CENTER FOR
Brains
Minds+
Machines

CBMM Memo No. 117 **June 15, 2022**

Deep Classifiers trained with the Square Loss

Mengjia Xu[1,2], **Akshay Rangamani**[1], **Andrzej Banburski**[1], **Qianli Liao**[1], **Tomer Galanti**[1], **Tomaso Poggio**[1]

[1]Center for Brains, Minds and Machines, MIT,
[2]Division of Applied Mathematics, Brown University

Abstract

Here we consider a simplified model of the dynamics of gradient flow under the square loss in ReLU networks. We study the convergence to a solution with the absolute minimum ρ, which is the product of the Frobenius norms of each layer weight matrix, when normalization by a Lagrange multiplier (LM) is used together with Weight Decay (WD). In the absence of LM + WD, good solutions for classification may still be achieved because of the implicit bias towards small norm solutions in the dynamics of gradient descent introduced by close-to-zero initial conditions on the norms of the weights. The main property of the minimizers that bounds their expected error is ρ: we prove that among all the close-to-interpolating solutions, the ones associated with smaller ρ have better margin and better bounds on the expected classification error. We also prove that quasi-interpolating solutions obtained by SGD in the presence of WD have a bias for low rank solutions; they also show the recently discovered behavior of Neural Collapse. Our analysis supports the idea that the advantage of deep networks relative to other standard classifiers is greater for the problems to which specific deep architectures such as CNNs can be applied. The deep reason is that CNNs reflect the function graph of certain locally compositional target function – which have small intrinsic dimensionality – and thus can be approximated well by sparse networks without incurring in the curse of dimensionality. Despite overparametrization the sparse networks show good generalization (because of the "small" covering numbers).

This is effectively an update of CBMM Memo 112, with some of the same material, more figures on experiments and a better discussion of generalization. Material which is interesting but less relevant, especially in the Appendices, is only in 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 in Deep Classifiers Trained with the Square Loss: Normalization, Low Rank, Neural Collapse and Generalization Bounds

Mengjia Xu[1,2], Akshay Rangamani[1], Andrzej Banburski[1], Qianli Liao[1], Tomer Galanti[1], Tomaso Poggio[1]

[1]Center for Brains, Minds and Machines, MIT
[2]Division of Applied Mathematics, Brown University

August 6, 2022

Abstract

Here we consider a simplified model of the dynamics of gradient flow under the square loss in ReLU networks. We study the convergence to a solution with the absolute minimum ρ, which is the product of the Frobenius norms of each layer weight matrix, when normalization by a Lagrange multiplier (LM) is used together with Weight Decay (WD) under forms of gradient descent. In the absence of LM + WD, good solutions for classification may still be achieved because of the implicit bias towards small norm solutions in the dynamics of gradient descent introduced by close-to-zero initial conditions on the norms of the weights. The main property of the minimizers that bounds their expected error is ρ: we prove that among all the close-to-interpolating solutions, the ones associated with smaller ρ have better margin and better bounds on the expected classification error. We also prove that quasi-interpolating solutions obtained by SGD in the presence of WD have a bias towards low rank solutions; they also enjoy the recently discovered behavior of Neural Collapse. Our analysis supports the idea that the advantage of deep networks relative to other standard classifiers is greater for the problems to which specific deep architectures such as CNNs can be applied. The deep reason is that CNNs reflect the function graph of certain locally compositional target function – which have small intrinsic dimensionality – and thus can be approximated well by sparse networks without incurring in the curse of dimensionality. Despite overparametrization the sparse networks show good generalization (because of the "small" covering numbers).

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 [1, 2], 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 [3, 1, 4]. Recently however, [5] has empirically demonstrated that it is possible to achieve a similar level of performance, if not better, using the square loss, paralleling older results for Support Vector Machines (SVMs) [6]. Can a theoretical analysis explain when and why regression should work well for classification? This question was the original motivation for this paper and preliminary versions of it [7].

In deep learning, unlike the case of linear networks, we expect from previous results (in the absence of regularization) several global minima with zero square loss, thus corresponding to interpolating solutions (in general degenerate, see [8, 9] and reference therein). Although all the 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 neither large margin nor good classification on a test set. When can we expect the solutions of the regression problem obtained by GD to have a large margin?

We introduce a simplified model of the training procedure that uses square loss, binary classification, gradient flow and Lagrange Multipliers (LN) for normalizing the weights. With this model we show that obtaining large margin interpolating solutions depends on the scale of initialization of the weights close to zero, in the absence of regularization (also called weight decay). Assuming convergence, we describe the qualitative dynamics of the deep network's parameters and show that ρ, which is the product of the Frobenius norms of the weight matrices, grows non-monotonically until small ρ, that is large margin, solutions are reached. Assuming that local minima and saddle points can be avoided, this analysis suggests that with small initialization and weight decay, gradient descent techniques may yield convergence to a minimum with a ρ biased to be as small as possible.
In the presence of weight decay, perfect interpolation of all data points cannot occur and is replaced by quasi-interpolation of the labels. In the special case of binary classification case in which $y_n = \pm 1$, quasi-interpolation is defined as $\forall\, n :\ |f(x_n) - y_n| \leq \epsilon$, where $\epsilon > 0$ is small. Our experiments and analysis of the dynamics show that, depending on the regularization parameter, there is a weaker dependence on initial conditions, as has been observed in [5]. We show that weight decay helps stabilize normalization of the weights, in addition to its role in the dynamics of the norm.
We then describe how to extend our model from gradient flow to an approximation of gradient descent. Next, the study of SGD reveals surprising differences relative to GD. In particular, in the presence of regularization, SGD does not converge to a perfect equilibrium: there is always, at least generically, SGD noise.The underlying reason is a rank constraint that depends on the size of the minibatches. This also implies a SGD bias towards small rank solutions that reinforces a similar bias due to maximization of the margin under normalization (that can be inferred from [10]).
Next we show that these quasi-interpolating solutions satisfy the recently discovered Neural Collapse (NC) phenomenon [11], assuming Stochastic Gradient Descent with minibatches. According to Neural Collapse, a dramatic simplification of deep network dynamics takes place – not only do all the margins become very similar to each other, 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 for the square loss under some assumptions.
Finally, we apply basic bounds on expected error to the solutions provided by SGD (for $\lambda > 0$), which have minimum ρ. The bounds though vacuous are much closer to the empirical test error than previous estimates. This is encouraging because in our setup large overparametrization, corresponding to interpolation of the training data, coexists with a relatively small Rademacher complexity.

Contributions The main contributions in this paper are

- We analyze the dynamics of deep network parameters, their norm, and the margins under gradient flow on the square loss, using *Lagrange normalization* (*LN*). We describe the evolution of ρ, and the role of Weight Decay and normalization in the training dynamics. We show a bias towards minimum ρ. The analysis is extended to the case of Gradient Descent in Section F.

- We show that when training the network using SGD with weight decay, there is always SGD noise, even asymptotically. In addition, there is a bias towards solutions of low rank.

- We show that critical points of SGD with weight decay and normalization that satisfy a certain pattern of interpolation and margins satisfy the conditions of Neural Collapse for deep networks trained with square loss.

- In our experiments with weight decay the generalization bounds are not far from being non-vacuous. In any case, they predict several qualitative observations on the relation between margin and generalization.

Outline The rest of the paper is organized as follows. We start by describing related work (section 2). In section 4 we formulate a model that assumes three simplifying assumptions. For this model we analyze the dynamics of gradient flow under the square loss for a binary classification problem. We use an analysis of its continuous dynamics to illuminate the role of Weight Decay and Weight Normalization in deep learning. We extend in Appendix F our model from Gradient Flow to an approximation of Gradient Descent. We then turn to analyze SGD, the associated noise and its bias towards low-rank solutions. In section 6 we use our analysis of the dynamics in the binary classification case to justify an assumption of margins being very close to each other at convergence to predict the phenomenon of

Neural Collapse when training on the square loss. A simple generalization bound links margin, which is the inverse of ρ, to expected error, in particular in the interpolation or quasi-interpolation case when the empirical error is zero or close to zero. Throughout we describe experiments on CIFAR10 that illustrate several of the theoretical results. We conclude in section 9 with a discussion of our results and their implications for generalization.

2 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 [1, 2, 13, 14]. Recent interest in using the square loss for classification has been spurred by the experiments in [5], though the practice of using the square loss is much older [6]. Muthukumar et. al. [15] recently showed for linear models that interpolating solutions for the square loss are equivalent to the solutions to the hard margin SVM problem (see also [7]). Recent work also studied interpolating kernel machines [16, 17] 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 [18, 19] 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 [20, 21, 22]. Detailed analyses of two-layer neural networks such as [23, 24, 25] typically use the square loss as an objective function. However these papers do not specifically consider the task of classification.
Several papers in recent years have studied the relationship between implicit regularization in linear neural networks and rank minimization. A main focus was on the matrix factorization problem, which corresponds to training a depth-2 linear neural network with multiple outputs w.r.t. the square loss (see references in [10]). Beyond factorization problems, it was shown that in linear networks of output dimension 1, gradient flow w.r.t. exponential-type loss functions converges to networks where the weight matrix of every layer is of rank 1. However, for nonlinear neural networks things are less clear. Empirically, several studies (see references in [10]) showed that replacing the weight matrices by low-rank approximations results in only a small drop in accuracy. This suggests that the weight matrices in practice are not too far from being low-rank.
Neural Collapse (NC) [11] 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 when training deep networks. Mixon et. al. [26] show NC in the regime of "unconstrained features". Recent results in [27] perform a more comprehensive analysis of NC in the unconstrained features paradigm. There have been a series of papers analytically showing the emergence of NC when using the cross-entropy loss [28, 29, 30]. In the study of the emergence of NC when training using the square loss, Ergen and Pilanci [31] (see also [32]) derived it through a convex dual formulation of deep networks. In addition to that, [33] and [34] show the emergence of NC in the unconstrained features regime. Our independent derivation is different from these approaches, and shows that NC emerges in the presence of normalization and weight decay.

3 Problem Setup

In this section, we describe the training settings considered in our work. We study training deep neural network with ReLU non-linearity using square loss minimization for classification problems. In the proposed analysis, we apply a specific normalization techniques: Weight Normalization, which is equivalent to Lagrange Multiplier, as well as regularization (also called Weight Decay), since such mechanisms seem commonly used for reliably training deep networks using gradient descent techniques [35, 5].

3.1 Assumptions

Throughout the theoretical analysis we make in some places simplifying assumptions relative to standard practice in deep neural networks. We mostly consider the case of binary classification though our analysis of Neural Collapse includes multiclass classification. We restrict ourselves to the square

loss. We consider gradient descent techniques but we assume different forms of them in various sections of the paper. In the first part, we assume continuous Gradient Flow (GF) instead of GD or Stochastic Gradient Descent (SGD). Gradient flow is the limit of discrete Gradient Descent algorithm with the learning rate being is infinitesimally small. In other sections, we describe an approimation of Gradient Descent within a Gradient Flow approach. SGD is specifically considered and shown to bias rank and induce asymtotic noise that is uniq to it. The analysis of Neural Collapse is carried out assuming GF on minibatches. Furthermore, we assume weight normalization by a Lagrange multiplier term added to the loss function, that normalizes the weight matrices. This is equivalent to Weight Normalization but is not equivalent to the more commonly used Batch Normalization.
We also assume throughout that the network is overparametrized and that that there is convergence to global minima with appropriate initialization, parameter values and data. A recent analysis [36] provide a powerful new criterion for convergence, assuming that the input dimension is greater than the number of data points.

3.2 Classification with Square Loss Minimization

In this work we consider a square loss minimization for classification along with regularization and weight normalization. We consider a binary classification problem given a training dataset $\mathcal{S} = \{(x_n, y_n)\}_{n=1}^N$, where $x_n \in \mathbb{R}^d$ are the inputs (normalized such that $\|x_n\| \leq 1$) and $y_n \in \{\pm 1\}$ are the labels. We use deep rectified homogeneous networks with L layers to solve this problem. For simplicity, we consider networks $f_W : \mathbb{R}^d \to \mathbb{R}^p$ of the following form $f_W(x) = W_L \sigma\left(W_{L-1} \ldots \sigma\left(W_1 x\right) \ldots\right)$, where $x \in \mathbb{R}^d$ is the input to the network and $\sigma : \mathbb{R} \to \mathbb{R}$, $\sigma(x) = \max(0, x)$ is the rectified linear unit (ReLU) activation function that is applied coordinate-wise at each layer. The last layer of the network is linear (see Figure 1).
Due to the positive homogeneity of ReLU (i.e., $\sigma(\alpha x) = \alpha\sigma(x)$ for all $x \in \mathbb{R}$ and $\alpha > 0$), one can reparametrize $f_W(x)$ by considering normalized[1] weight matrices $V_k = \frac{W_k}{\|W_k\|}$ and define $\rho_k = \|W_k\|$ obtaining $f_W(x) = \rho_L V_L \sigma\left(\rho_{L-1} \ldots \sigma\left(\rho_1 V_1 x\right) \ldots\right)$. Because of homogeneity of the ReLU it is possible to pull out the product of the layer norms as $\rho = \prod_k \rho_k$ and write $f_W(x) = \rho f_V(x) = \rho V_L \sigma\left(V_{L-1} \ldots \sigma\left(V_1 x\right) \ldots\right)$. Notice that the two networks – $f_W(x)$ and $\rho f_V(x)$ – are equivalent reparameterizations of the same function (if $\rho = \prod_k \rho_k$) but their optimization generally differ. We define $f_n := f_V(x_n)$
We adopt in our definition the convention that the norm ρ_j of the convolutional layers is *the norm of their filters* and not the norm of their associated Toeplitz matrices. The reason is that the calculation of the covering numbers must take into account the associated shift invariance (see section 3.3 in [37] and [38]). The ρ_j calculated in this way is the quantity that enters the generalization bounds of section 8.
In practice, certain normalization techniques are used in order to train neural networks. This is usually performed using either batch normalization (BN) or, less frequently, weight normalization (WN). BN consists of standardizing the output of the units in each layer to have zero mean and unit variance. WN normalizes the weight matrices (section 10 in [4]). In our analysis, we model normalization by normalizing the weight matrices, using a *Lagrange Multiplier* (*LN*) term added to the loss function. This approach is equivalent to WN.
In the presence of normalization, we assume that all layers are normalized, except for the last one, via the added Lagrange multiplier. Thus, the weight matrices $\{V_k\}_{k=1}^L$ are constrained by the Lagrange multiplier term to be close to, and eventually converge to, unit norm matrices (in fact to fixed norm matrices); notice that normalizing V_L and then multiplying the output by ρ, is equivalent to letting $W_L = \rho V_L$ be unnormalized. Thus, f_V is the network that at convergence has $L-1$ normalized layers (see Figure 1).
Minimization of the regularized loss function under the constraint $\|V_k\|^2 = 1$ can be summarized in the following manner

$$\begin{aligned} \mathcal{L}_{\mathcal{S}}(\rho, \{V_k\}_{k=1}^L) : &= \frac{1}{N} \sum_n (\rho f_n - y_n)^2 + \sum_{k=1}^L \nu_k (\|V_k\|^2 - 1) + \lambda \rho^2 \\ &= \frac{1}{N} \sum_n (1 - \rho \bar{f}_n)^2 + \sum_{k=1}^L \nu_k (\|V_k\|^2 - 1) + \lambda \rho^2, \end{aligned} \tag{1}$$

[1] We choose the Frobenius norm here to simplify our calculations.

where ν_k are the Lagrange multipliers and $\lambda > 0$ is a predefined parameter.

Separability and Margins. Two important aspects of classification are *separability* and *margins*. For a given sample (x, y) (train or test sample) and model f_W, we say that f_W correctly classifies x, if $\bar{f}_n = y_n f_n > 0$. In addition, for a given dataset $\mathcal{S} = \{(x_n, y_n)\}_{n=1}^N$, *separability* is defined as the condition in which all training samples are classified correctly, $\forall\, n \in [N] :\ \bar{f}_n > 0$. Furthermore, when $\sum_{n=1}^N \bar{f}_n > 0$, we say that *average separability* is satisfied. The minimum of $\mathcal{L}_\mathcal{S}$ for $\lambda = 0$ is usually zero under our assumption of overparametrization. This corresponds to separability.
Notice that if f_W is a zero loss solution of the regression problem, then $\forall\, n :\ f_W(x_n) = y_n$, which is also equivalent to $\rho f_n = y_n$, where we call $y_n f_n = \bar{f}_n$ *the margin*[2] *for* x_n. By multiplying both sides of this equation by y_n, and summing both sides over $n \in [N]$, we obtain that $\rho \sum_n \bar{f}_n = N$. Thus, the norm ρ of a minimizer is inversely proportional to its average margin μ in the limit of $\lambda = 0$, with $\mu = \frac{1}{N}\sum_n \bar{f}_n$. It is also useful to define *the margin variance* $\sigma^2 = M - \mu^2$ *with* $M = \frac{1}{N}\sum_n \bar{f}_n^2$. Notice that $M = \frac{1}{N}\sum_n \bar{f}_n^2 = \sigma^2 + \mu^2$ and that both M and σ^2 are not negative.

Interpolation and Quasi-interpolation. Assume that the weights V_k are normalized at convergence. Then

Lemma 1 *In the absence of regularization, there are solutions that interpolate all data points with the same margin and achieve zero loss. For $\lambda > 0$ there are no solutions that have the same margins and interpolate. However there are solutions with the same margins that quasi-interpolate and are critical points of the gradient.*

Proof Consider the loss $\mathcal{L}_\mathcal{S} = \frac{1}{N}\sum_n (1-\rho\bar{f}_n)^2 + \lambda\rho^2 = 1 - 2\rho\mu + \rho^2 M + \lambda\rho^2$. For $\lambda = 0$, a zero of the loss $\mathcal{L}_\mathcal{S} = 0$ implies $\forall\, n \in [N] :\ \mu = \bar{f}_n$ and $\mu = \frac{1}{\rho}$. However, for $\lambda > 0$ the assumption that all $\bar{f}_n$ are equal yields $M = \mu^2$ and thus $\mathcal{L}_\mathcal{S} = \rho^2\mu^2 - 2\rho\mu + (1 + \lambda\rho^2)$. Setting $\mathcal{L}_\mathcal{S} = 0$ gives a second order equation in ρ which does not have real-valued solutions for $\lambda > 0$. Thus in the presence of regularization, there exist no solutions that have the same margin for all points and reach zero empirical loss. However, solutions that have the same margin for all points and correspond to zero gradient w.r.t. ρ exist. To see this, assume $\sigma = 0$, setting the gradient of $\mathcal{L}_\mathcal{S}$ w.r.t. ρ equal to zero, yielding $\rho\mu^2 - \mu + \lambda\rho = 0$. This gives $\rho = \frac{\mu}{\mu^2 + \lambda}$. This solution yields $\rho\mu < 1$, which corresponds to non-interpolating solutions. ■

Experiments We performed binary classification experiments using the standard CIFAR10 dataset [39]. Image samples with class labels 1 and 2 were extracted for the binary classification task. The total number of training and test data points are 10000 and 2000, respectively. The model architecture in Fig. 1b contains four convolutional layers, two fully connected layers with hidden sizes 1024 and 2. The number of channels for the four convolutional layers are 32, 64, 128 and 128, the filter size is 3×3. The first fully connected layer has $3200 \times 1024 = 3,276,800$ weights and the very last layer has $1024 \times 2 = 2048$ weights. At the top layer of our model, there is a learnable parameter ρ (Fig. 1b). Following each convolution layer, we applied a ReLU nonlinear activation function and Weight Normalization (WN) to all layers, freezing the weights of the WN parameter "g" in the Weight Normalization algorithm and normalizing the $\{V_k\}_{k=1}^{L-1}$ matrices at each layer w.r.t. their Frobenius norm, while the top layer's norm is denoted by ρ.

4 Training Dynamics

4.1 Gradient Flow Equations

The gradient flow equations are as follows

$$\begin{aligned} \dot{\rho} &= -\frac{\partial \mathcal{L}_\mathcal{S}(\rho, \{V_k\}_{k=1}^L)}{\partial \rho} = \frac{2}{N}\sum_n (1 - \rho\bar{f}_n)\bar{f}_n - 2\lambda\rho \\ \dot{V}_k &= -\frac{\partial \mathcal{L}_\mathcal{S}(\rho, \{V_k\}_{k=1}^L)}{\partial V_k} = \frac{2}{N}\sum_n (1 - \rho\bar{f}_n)\rho\frac{\partial \bar{f}_n}{\partial V_k} - 2\nu_k V_k. \end{aligned} \tag{2}$$

[2]Notice that the term "margin" is usually defined as $\min_{n\in[N]} \bar{f}_n$. Instead, we use the term "margin for x_n" to distinguish our definition from the usual one.

In the second equation we can use the unit norm constraint on the $\|V_k\|$ to determine the Lagrange multipliers ν_k. Using a structural property of the gradient[3], the constraint $\|V_k\|^2 = 1$ implies $\frac{\partial \|V_k\|^2}{\partial t} = V_k^T \dot{V}_k = 0$, which gives

$$\nu_k = \frac{1}{N}\sum_n (\rho \bar{f}_n - \rho^2 f_n^2) = \frac{1}{N}\sum_n \rho \bar{f}_n (1 - \rho f_n). \tag{4}$$

Thus the gradient flow is the following dynamical system

$$\dot{\rho} = \frac{2}{N}\left[\sum_n \bar{f}_n - \sum_n \rho(\bar{f}_n)^2\right] - 2\lambda\rho \text{ and } \dot{V}_k = \frac{2}{N}\rho \sum_n \left[\left(1 - \rho \bar{f}_n\right)\left(-V_k \bar{f}_n + \frac{\partial \bar{f}_n}{\partial V_k}\right)\right] \tag{5}$$

In particular, we can also write

$$\dot{\rho} = 2(\mu - \rho(M + \lambda)), \tag{6}$$

hence, at critical points (when $\dot{\rho} = 0$ and $\dot{V}_k = 0$), we have

$$\rho = \rho_{\text{eq}} := \frac{\frac{1}{N}\sum_n \bar{f}_n}{\lambda + \frac{1}{N}\sum_n \bar{f}_n^2} = \frac{\sum_n \bar{f}_n}{\lambda + \sum_n \bar{f}_n^2} = \frac{\mu}{M + \lambda}. \tag{7}$$

Thus the gap to interpolation due to $\lambda > 0$ is $\epsilon = (\rho_{\lambda=0} - \rho_\lambda)\mu = 1 - \frac{\mu}{M+\lambda}\mu$ which gives

$$\epsilon = \frac{\mu^2}{\mu^2 + \sigma^2 + \lambda}. \tag{8}$$

Notice that since the V_k are bounded functions they must take their maximum and minimum values on their compact domain – the sphere – because of the extremum value theorem. Also notice that for normalized V_k, $V_k^T \dot{V}_k = 0$ always, that is for normalized V_k the change in V_k is always orthogonal to V_k, that is V_k can only rotate. If $\dot{V}_k = 0$ then the weights V_k are given by[4]

$$V_k = \frac{\sum_n \ell_n \frac{\partial f_n}{\partial V_k}}{\sum \ell_n f_n}, \tag{9}$$

where $\ell_n = 1 - \rho \bar{f}_n$.
As a consequence of the above derivation, we can represent the loss function in terms of ρ, $\dot{\rho}$ and μ in the following manner.

Lemma 3 *Let $f_W(x) = \rho f_V(x)$ be a neural network, with $\forall\ k \in [L]:\ \|V_k\| = 1$. The square loss can be written as $\mathcal{L}_S(\rho, \{V_k\}_{k=1}^L) = 1 - \rho(\frac{1}{2}\dot{\rho} + \mu)$*

Proof First, we consider that

$$\begin{aligned}
\mathcal{L}_S(\rho, \{V_k\}_{k=1}^L) &= \frac{1}{N}\sum_n (\rho f_n - y_n)^2 + \sum_{k=1}^L \nu_k(\|V_k\|^2 - 1) + \lambda\rho^2 \\
&= \frac{1}{N}(\rho^2 f_n^2 - 2y_n\rho f_n + y_n^2) + \lambda\rho^2 \\
&= 1 - 2\rho\mu + \rho^2 M + \lambda\rho^2,
\end{aligned}$$

where the second equations follows from $\forall\ k \in [L]:\ \|V_k\| = 1$ and the third from $y_n^2 = 1$, $\mu = \sum_n y_n f_n$ and $M = \sum_n f_n^2$. On the other hand, by Equation 6, $\dot{\rho} = 2\mu - 2\rho M - 2\lambda\rho$ which gives $2\rho M = 2\mu - 2\lambda\rho - \dot{\rho}$. Therefore, we conclude that $\mathcal{L}_S(\rho, \{V_k\}_{k=1}^L) = 1 - \frac{1}{2}\rho\dot{\rho} - \rho\mu = 1 - \rho(\frac{1}{2}\dot{\rho} + \mu)$ as desired. ■

[3] Let $f_W(x)$ be a ReLU neural network. The following structural property of the gradient holds.

Lemma 2 (Lemma 2.1 of [40]) *Let $f_W(x) = W_L\sigma(W_{L-1}\ldots\sigma(W_1 x)) : \mathbb{R}^d \to \mathbb{R}$ be a ReLU neural network. Then, we can write:*

$$\forall x \in \mathbb{R}^d:\ \sum_{i,j} \frac{\partial f_W(x)}{\partial W_k^{i,j}} W_k^{i,j} = \left\langle W_k, \frac{\partial f_W(x)}{\partial W_k} \right\rangle = f_W(x). \tag{3}$$

[4] This overdetermined system of equations – with as many equations as weights – can also be used to reconstruct the training set from the V_k, the y_n and the f_n.

Convergence. A favorable property of optimization of the square loss is the convergence of the relevant parameters. With gradient descent, the loss function cannot increase, while the trainable parameters may potentially diverge. A typical scenario of this kind happens with cross-entropy minimization, where the weights typically tend to infinity. In light of the theorems in Section 4.2, we could hypothetically think of training dynamics in which the loss function's value $\mathcal{L}(\rho, \{V_k\}_{k=1}^L)$ decreases while ρ oscillates periodically within some interval. As we show next, this is impossible when the loss function's value converges to zero.

Lemma 4 *Let $f_W(x) = \rho f_V(x)$ be a neural network and $\lambda = 0$. Assume that during training time, we have $\lim_{t\to\infty} \mathcal{L}(\rho, \{V_k\}_{k=1}^L) = 0$ and $\forall\ k \in [L] :\ \|V_k\| = 1$. Then, ρ and V_k converge (i.e., $\dot{\rho} \to 0$ and $\dot{V}_k \to 0$).*

Proof Note that if $\lim_{t\to\infty} \mathcal{L}(\rho, \{V_k\}_{k=1}^L) = 0$, then, for all $n \in [N]$, we have: $(\rho f_n - y_n)^2 \to 0$. In particular, $\rho f_n \to y_n$ and $\rho \bar{f}_n \to 1$. Hence, we conclude that $\mu\rho \to 1$. Therefore, by Lemma 3, $\rho\dot{\rho} \to 0$. We note that $\rho \to 0$ would imply $\rho f_n \to 0$ which contradicts $\mathcal{L}(\rho, \{V_k\}_{k=1}^L) \to 0$, since the labels y_n are non-zero. Therefore, we conclude that $\dot{\rho} \to 0$. To see why $\dot{V}_k \to 0$, we recall that

$$\dot{V}_k = \frac{2}{N}\rho \sum_n \left[\left(1 - \rho \bar{f}_n\right) \left(-V_k \bar{f}_n + \frac{\partial \bar{f}_n}{\partial V_k} \right) \right].$$

We note that $\|V_k\| = 1$, $|\bar{f}_n| = 1$ and $\frac{\partial \bar{f}_n}{\partial V_k}$ is bounded (assuming that $\forall\ n \in [N] :\ \|x_n\| \leq 1$ and $\forall\ k \in [L] :\ \|V_k\| = 1$). Hence, since ρ converges, $\rho \bar{f}_n \to 1$, implying (for $\lambda = 0$) $\dot{V}_k \to 0$. ■

4.2 Landscape of the empirical risk

As a next step, we establish key properties of the loss landscape. The landscape of the empirical loss contains a set of degenerate zero-loss global minima (for $\lambda = 0$) that under certain overparametrization assumptions may be connected in a single zero-loss degenerate valley for $\rho \geq \rho_0$. Figure 2 shows a landscape which has a saddle for $\rho = 0$ and then goes to zero loss (zero crossing level, that is the coastline) for different values of ρ (look at the boundary of the mountain). As we will see in our analysis of the gradient flow, the descent from $\rho = 0$ can encounter local minima and saddles with non-zero loss. Furthermore, even though the valley of zero loss may be connected, the absolute minimum ρ point may be unreachable by gradient flow from another point of zero loss even in the presence of $\lambda > 0$, because of the possible non-convex profile of the coastline (see inset of Figure).
The degeneracy of the global minimizers achieving zero loss of the unregularized loss function $\mathcal{L}_S = \sum_{i=1}^n (f_W(x_i) - y_i)^2$ was established recently under the assumption that the network f is over-parametrized with a number of weights $d \gg n$ and that the network is able to interpolate the training data, achieving $\min_W \mathcal{L}(f_W) = 0$ which implies $\forall i \in [n] :\ f_W(x_i) = y_i$.
If we assume overparametrized networks with $d \gg n$, where d is the number of parameters and n is the number of data points [9] proved that the global minima of $L(w)$ are highly degenerate[5] with dimension $d - n$.

Theorem 1 ([41], informal) *We assume an overparametrized neural network f_W with smooth ReLU activation functions and square loss. Then the minimizers W^* achieve zero loss and are highly degenerate with dimension $d - n$.*

Furthermore, for "large" overparametrization, all the global minima – associated to interpolating solutions – are connected within a unique, large valley. The argument is based on Theorem 5.1 of [42]:

Theorem 2 ([42], informal) *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.*

[5]This result is also what one expects from Bezout theorem for a deep polynomial network. As mentioned in Terry Tao's blog "from the general "soft" theory of algebraic geometry, we know that the algebraic set V is a union of finitely many algebraic varieties, each of dimension at least d-n, with none of these components contained in any other. In particular, in the under-determined case $n < d$, there are no zero-dimensional components of V , and thus V is either empty or infinite"(see references in [41]).

In particular, the theorem implies that *zero-square-loss minima with different values of ρ are connected*. A connected single valley of zero loss *does not* however guarantee that *SGD with WD will converge to the global minimum which is now > 0, 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 may not monotonically increase or decrease ρ.
For large ρ (mainly in the case $\lambda = 0$) we expect many solutions. The existence of several solutions for large ρ is based on the following intuition: the last linear layer is enough – if the layer before the linear classifier has more units than the number of training points – to provide solutions for a given set of random weights in the previous layers (for large ρ and small f_i). This also means that 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 ρ smaller than a certain minimum ρ, which is network and data-dependent. With increasing ρ from $\rho = 0$ there will be a continuous set of zero square-loss degenerate minima with the minimizer representing an interpolating (for $\lambda = 0$) or almost interpolating solution (for $\lambda > 0$). We expect that $\lambda > 0$ results in a "pull" towards the minimum ρ_0 within the local degenerate minimum of the loss.

4.3 Qualitative Dynamics

Recall that $\forall\, n \in [N] :\ 0 \leq |\bar{f}_n| \leq 1$ because the assumption $\|x\| \leq 1$, yields $\|f(x)\| \leq 1$ by taking into account the definition of ReLUs and the fact that matrix norms are sub-multiplicative. Depending on the number of layers, the maximum margin that the network can achieve for a given dataset is usually much smaller than the upper bound 1, because the weight matrices have unit norm and the bound ≤ 1 is conservative. Thus, in order to guarantee interpolation, namely, $\rho f_n y_n = 1$, ρ must be significantly larger than 1. For instance, in the experiments plotted in this paper, the maximal $\bar{f}_n$ is ≈ 0.002 and thus the ρ needed for interpolation (for $\lambda = 0$) is in the order of 500. We assume then that for a given dataset there is a maximal value of $y_n f_n$ that allows interpolation. Correspondingly, there is a minimum value of ρ that we call, as mentioned earlier, ρ_0.
We now provide some intuition for the dynamics of the model. Notice that $\rho(t) = 0$ and $f_V(x) = 0$ (if all weights are zero) is a critical unstable point. A small perturbation will either result in $\dot{\rho} < 0$ with ρ going back to zero or in ρ growing if the average margin is just positive, that is $\mu > \lambda\rho > 0$.

Small ρ initialization. First, we consider the case where the neural network is initialized with a smallish ρ, that is $\rho < \rho_0$. Assume then that at some time t, $\mu > 0$, that is *average separability* holds. Notice that if the f_n were zero-mean, random variables, there would be a 50% chance for average separability to hold. Then Equation (5) shows that $\dot{\rho} > 0$. If full separability takes place, that is $\forall\, n :\ f_n > 0$, then $\dot{\rho}$ remains positive at least until $\rho = 1$. This is because Equation (5) implies that $\dot{\rho} \geq 2(\mu - \rho\mu)$ since $M \leq \mu$. In general, assuming eventual convergence, ρ may grow non-monotonically, that is there may oscillations in ρ for "short" intervals, until it converges to ρ_0.
Following Lemma 3, if $\dot{\rho}$ becomes negative during training, then, the average margin μ must increase since GD cannot increase but only decrease $\mathcal{L}$. In particular, this implies that $\dot{\rho}$ cannot be negative for long periods of time. Notice that short periods of decreasing ρ are "good" since they increase the average margin!
If $\dot{\rho}$ turns negative, it means that it has crossed $\dot{\rho} = 0$. This may be a critical point for the system if the values of V_k corresponding to $\dot{V}_k = 0$ are compatible (since the matrices $\{V_k\}_{k=1}^{L}$ determine the value of $\bar{f}_n$). We assume that this critical point – either a local minimum or a saddle – can be avoided by the randomness of SGD or by an algorithm that restarts optimization when a critical point is reached for which $\mathcal{L} > 0$.
Thus, ρ grows (non-monotonically) until it reaches an equilibrium value, close to ρ_0. Recall that for $\lambda = 0$ this corresponds to a degenerate global minimum $\mathcal{L} = 0$, usually resulting in a large attractive basin in the loss landscape (see Appendix 4.2). For $\lambda = 0$, a zero value of the loss $\mathcal{L} = 0$ implies interpolation: thus all the f_n have the same value, that is all the margins are the same. Notice that all ρ_k of Figure 1 are basically constant because of weight normalization, see Figure 3.

Large ρ initialization. If we initialize a network with large norm $\rho > \rho_0$, Equation 1 shows that $\dot{\rho} < 0$. This implies that the norm of the network will decreases until eventually an equilibrium is reached. In fact since $\rho \gg 1$ it is likely that there exists an interpolating (or near interpolating) solution with ρ that

is very close to the initialization. In fact, for large ρ it is usually empirically possible to find a set of weights V_L such that $\rho\bar{f}_n \approx 1$. To understand why this may be true, recall that if there are at least N units in the top layer of the network (layer L) with given activities and $\rho \gg \rho_0$ there exist values of V_L that yield interpolation due to Theorem 2. In other words, it is easy for the network to interpolate with small values $\bar{f}_n$. These large ρ, small $\bar{f}_n$ solutions are reminiscent of the Neural Tangent Kernel (NTK) solutions [22], where the parameters do not move too far from their initialization. A formal version of the same argument is based on the following result.
We now assume that the network in the absence of weight decay has converged to an interpolating solution

Lemma 5 *Let f_V be a neural network with weights $\{V_k\}_{k=1}^L$, such that, $\forall\, n \in [N]:\ \rho\bar{f}_n = \rho\mu^* = 1$. Further assume that the classifier V_L and the last layer features h are aligned, ie, $y_n\langle V_L, h(x_n)\rangle = ||h(x_n)||_2$, where the vector h denotes the activities of the units in the last layer. Then, perturbing V_L into another unit-norm vector $V_L' \in \mathbb{R}^p$, such that, $V_L^T V_L' = \alpha \in (0,1)$ yields a neural network $\hat{f}(x) = \langle V_L', h(x)\rangle$ with the property that $\frac{\rho}{\alpha}\hat{f}$ is an interpolating solution, corresponding to a critical point of the gradient but with a larger ρ.*

Proof Consider the margins of the network $\hat{f}(x) = \langle V_L', h(x)\rangle$, we have that $\bar{\hat{f}}_n = y_n\langle V_L', h(x_n)\rangle$. Since the classifier weights and the last layer features are aligned (as it may happen for $\lambda \to 0$), we have that $y_n h(x_n) = ||h(x_n)|| \times V_L$. This means $\bar{\hat{f}}_n = ||h(x_n)|| \times \langle V_L', V_L\rangle$. We also have from the interpolating condition that $\rho\bar{f}_n = \rho\mu^* = 1$, which means $||h(x_n)|| = \frac{1}{\rho}$. Putting all this together, we have $\frac{\rho}{\alpha}\bar{\hat{f}}_n = 1$, which concludes the proof. ■

Thus if a network exists providing an interpolating solution with a minimum ρ and $V_L \propto h$, there exist networks, that differ only in the last V_L layer, that are also interpolating but with larger ρ. As a consequence there is a continuum of solutions that differ only in the weights V_L of the last layer.
Of course there may be interpolating solutions corresponding to different sets of weights in layers below L, to which the above statement does not apply. These observations suggest that there is a valley of minimizers for increasing ρ, starting from a zero-loss minimizer which have the neural collapse property (see section 6).
In Figure 4 we show the dynamics of ρ alongside train loss and test error. We show results with and without Weight Decay in the top and bottom rows of Figure 4 respectively. $\mathcal{L}_\mathcal{S}$ decreases with μ increasing and σ decreasing. The figures show that in our experiments the large margins of some of the data points decrease during GD, contributing to a decrease in σ. Furthermore Equation (4.1) suggests that for small ρ, the term dominating the decrease in $\mathcal{L}_\mathcal{S}$ is $-2\rho\mu$. For larger ρ, the term $\rho^2 M = \rho^2(\sigma^2+\mu^2)$ becomes important: eventually $\mathcal{L}_\mathcal{S}$ decreases because σ^2 decreases. The regularization term, for standard small values of λ, is relevant only in the final phase, when ρ is in the order of ρ_0. For $\lambda = 0$ the loss at the global equilibrium (which happens at $\rho = \rho_0$) is $\mathcal{L}_\mathcal{S} = 0$ (since $\mu = \frac{1}{\rho_0}$, $M = \mu^2$, $\sigma^2 = 0$).
To sum up, starting from small initialization, gradient techniques will explore critical points with ρ growing from zero. Thus quasi-interpolating solutions with small ρ (corresponding to large margin solutions) may be found before the many large ρ quasi-interpolating solutions which have worse margins (see Figure 4, upper and lower row). This dynamics can take place *even in the absence of regularization*; however, $\lambda > 0$ makes the process more robust and bias it towards small ρ.

5 The origin of SGD noise and a bias towards low-rank weight matrices

In the previous sections we assumed that ρ and V_k are trained using GF. In this section we consider a slightly different setting where SGD is applied instead of GF. Specifically, V_k and ρ are first initialized and then iteratively updated simultaneously in the following manner

$$\rho \leftarrow \rho - \eta \cdot \frac{\mathcal{L}_{S'}(\rho, \{V_k\}_{k=1}^L)}{\partial\rho} \text{ and } V_k \leftarrow V_k - \eta \cdot \frac{\mathcal{L}_{S'}(\rho, \{V_k\}_{k=1}^L)}{\partial V_k}, \tag{10}$$

where $\mathcal{S}'$ is selected uniformly as a subset of $\mathcal{S}$ of size B and $\eta > 0$ is the learning rate.
An intriguing argument for small rank weight matrices is the following observation that follows from Equation (5) (see also [7]).

Lemma 6 *Let f_W be a neural network. Assume that we iteratively train ρ and $\{V_k\}_{k=1}^L$ using the process described above with weight decay $\lambda > 0$. Suppose that training converges, that is $\frac{\partial \mathcal{L}_{\mathcal{S}'}(\rho, \{V_k\}_{k=1}^L)}{\partial \rho} = 0$ and $\forall\, k \in [L]:\ \frac{\partial \mathcal{L}_{\mathcal{S}'}(\rho, \{V_k\}_{k=1}^L)}{\partial V_k} = 0$ for all mini-batches $\mathcal{S}' \subset \mathcal{S}$ of size $B < |\mathcal{S}|$. Assume that $\forall\, n \in [N]:\ \bar{f}_n \neq 0$. Then, the ranks of the matrices V_k are at most ≤ 2.*

Proof We would like to show that the matrix $\frac{\partial f_W(x)}{\partial V_k}$ is of rank ≤ 1. We note that for any input $x \in \mathbb{R}^n$, the output can be written as follows, $f_V(x) = V_L \cdot D_{L-1}(x) \cdots D_1(x) \cdot V_1 \cdot x$, where $D_i(x) = \text{diag}[\sigma'(u_i(x)))]$ and $u_i(x) = W_i \cdot D_{L-1}(x) \cdots D_1(x) \cdot V_1 \cdot x$. With measure 1 over the selection of W, the matrices $\{D_l(x)\}_{l=1}^{L-1}$ are constant in the neighborhood of W. Let $a^\top = V_L \cdot D_{L-1}(x) \cdots D_k(x)$ and $b = D_{k-1}(x) \cdot W_{k-1} \cdots W_1 x$. We can write $f_V(x) = a(x)^\top \cdot V_k \cdot b(x)$. Since the derivatives of a and b with respect to V_k are zero and a, b are vectors, we have $\frac{\partial \bar{f}_n}{\partial V_k} = y_n \cdot \frac{\partial f_V(x_n)}{\partial V_k} = y_n \cdot a(x_n) \cdot b(x_n)^\top$ which is a matrix of rank ≤ 1.
Since $\forall\, k \in [L]:\ \frac{\partial \mathcal{L}_{\mathcal{S}'}(\rho, \{V_k\}_{k=1}^L)}{\partial V_k} = 0$ for all mini-batches $\mathcal{S}' = \{(x_{i_j}, y_{i_j})\}_{j=1}^B \subset \mathcal{S}$ of size $B < |\mathcal{S}|$, we have

$$\frac{\partial \mathcal{L}_{\mathcal{S}'}(\rho, \{V_k\}_{k=1}^L)}{\partial V_k} = \frac{2}{B}\rho \sum_{j=1}^{B} \left[\left(1 - \rho \bar{f}_{i_j}\right) \left(-V_k \bar{f}_{i_j} + \frac{\partial \bar{f}_{i_j}}{\partial V_k} \right) \right] = 0. \tag{11}$$

Since interpolation is impossible when training with $\lambda > 0$, there exists at least one $n \in [N]$ for which $\rho \bar{f}_n \neq 1$. We consider two batches $\mathcal{S}'_i$ and $\mathcal{S}'_j$ of size B that differ by sample, (x_i, y_i) and (x_j, y_j). We have

$$\begin{aligned} \forall\, i, j \in [N]:\ 0 &= \frac{\partial \mathcal{L}_{\mathcal{S}'_i}(\rho, \{V_k\}_{k=1}^L)}{\partial V_k} - \frac{\partial \mathcal{L}_{\mathcal{S}'_j}(\rho, \{V_k\}_{k=1}^L)}{\partial V_k} \\ &= \frac{2}{B} \cdot \rho \left[\left(1 - \rho \bar{f}_i\right) \left(-V_k \bar{f}_i + \frac{\partial \bar{f}_i}{\partial V_k} \right) - \left(1 - \rho \bar{f}_j\right) \left(-V_k \bar{f}_j + \frac{\partial \bar{f}_j}{\partial V_k} \right) \right]. \end{aligned} \tag{12}$$

Assume that there exists a pair $i, j \in [N]$ for which $(1 - \rho \bar{f}_i)\bar{f}_i \neq (1 - \rho \bar{f}_j)\bar{f}_j$. Then, we can write

$$V_k = \frac{\left[(1 - \rho \bar{f}_i) \cdot \frac{\partial \bar{f}_i}{\partial V_k} + (1 - \rho \bar{f}_j) \cdot \frac{\partial \bar{f}_j}{\partial V_k} \right]}{[(1 - \rho \bar{f}_i)\bar{f}_i - (1 - \rho \bar{f}_j)\bar{f}_j]}. \tag{13}$$

Since $\frac{\partial \bar{f}_i}{\partial V_k}$ and $\frac{\partial \bar{f}_j}{\partial V_k}$ are matrices of rank ≤ 1, we obtain that V_k is of rank ≤ 2. Otherwise, assume that for all pairs $i, j \in [N]$, we have $\alpha = (1 - \rho \bar{f}_i)\bar{f}_i = (1 - \rho \bar{f}_j)\bar{f}_j$. In this case we obtain that for all $i, j \in [N]$, we have

$$\left(1 - \rho \bar{f}_i\right) \cdot \frac{\partial \bar{f}_i}{\partial V_k} = \left(1 - \rho \bar{f}_j\right) \cdot \frac{\partial \bar{f}_j}{\partial V_k} = U. \tag{14}$$

Therefore, since $\alpha = (1 - \rho \bar{f}_i)\bar{f}_i = (1 - \rho \bar{f}_j)\bar{f}_j$, by Equation 11,

$$0 = \frac{2}{B}\rho \sum_{j=1}^{B} \left[\left(1 - \rho \bar{f}_{i_j}\right) \left(-V_k \bar{f}_{i_j} + \frac{\partial \bar{f}_{i_j}}{\partial V_k} \right) \right] = -2\rho\alpha V_k + 2\rho U. \tag{15}$$

Since the network cannot perfectly fit the dataset when trained with $\lambda > 0$, we obtain that there exists $i \in [N]$ for which $(1 - \rho \bar{f}_i) \neq 0$. Since $\bar{f}_i \neq 0$ for all $i \in [N]$, this implies that $\alpha \neq 0$. We conclude that V_k is proportional to U which is of rank ≤ 1. ■

5.1 Origin of SGD noise

Lemma 6 shows that there cannot be convergence to a unique set of weights $\{V_k\}_{k=1}^L$ that satisfy equilibrium for all minibatches. More details of the argument are illustrated in [43]. When $\lambda = 0$, interpolation of all data points is expected: in this case, the GD equilibrium can be reached without any constraint on the weights. This is also the situation in which SGD noise is expected to essentially disappear: compare the histograms on the left and the right hand side of Figure 5.
Thus, during training, the solution $\{V_k\}_{k=1}^L$ is not the same for all samples: there is *no convergence to a unique solution* but instead attempts to "jump" from one to another during training.

The absence of convergence to a unique solution is not surprising for SGD when the landscape is not convex[6] [7].

5.2 Margins variance

Unregularized square loss minimization in which we have interpolation of all labels implies that all margins are the same ($\sigma^2 = \sum_n (\bar{f}_n - \sum \bar{f}_n)^2) = 0$). We have shown however that in SGD, differently from GD, there is SGD noise. This is equivalent to saying that that SGD does not converge to a unique solution that corresponds to zero gradient for all data point. This also means that there is variance in the values of f across different minibatches and correspondingly for the associated values of ρ.
To have an intuition of the order of magnitude of the fluctuations in the margin that may be expected consider the following simple case: assume that for one specific minibatch of size B the network would exactly interpolate with all the same B margins $\bar{f}_n = \frac{1}{\rho_{eq}}$ if $\lambda = 0$; for the same B data points but with $\lambda > 0$ the network will quasi-interpolate with $\rho = \frac{\mu}{B+\lambda}$ and with the same margins for all the B datapoints.
In this example the gap to interpolation is the same ϵ which is relevant for Neural Collapse (see later assumption 2). If, in this binary case, we consider margins separately for the $+$ and $-$ one-hot encodings, as in the multiclass case (see later), then $1 > f^+(x_+) > 1 - \epsilon$ and $\epsilon > f^+(x_-) > 0$, with ϵ and fluctuations around it (for individual data points). Figure 5 shows that the variance of the noise depends on λ and becomes very small when $\lambda = 0$ (for the square loss).

6 Neural Collapse

A recent paper [11] 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 progressively decreases. Direct empirical measurements expose an inductive bias they call Neural Collapse (NC), involving four interconnected phenomena. Informally, (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 decision rule). In this section we show that we can derive the conditions of Neural Collapse by studying the dynamics of Stochastic Gradient Descent on the square loss function with Lagrange Normalization and Weight Decay.
We now formally define the four neural collapse conditions. We consider a network $f_W(x) = W_L h(x)$, where $h(x) \in \mathbb{R}^p$ denotes the last layer feature embedding of the 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 $\mathcal{S} = \{(x_{cn}, y_{cn})\}_{n=1,c=1}^{N,C}$ with N samples per class. We can compute the per-class mean of the last layer features as follows

$$\mu_c = \frac{1}{N} \sum_{n=1}^{N} h(x_{cn}), \tag{16}$$

The global mean of all features as follows

$$\mu_G = \frac{1}{C} \sum_c \mu_c = \frac{1}{NC} \sum_{c=1,n=1}^{C,N} h(x_{cn}).$$

[6] Suppose that the loss landscape has two equivalent minima separated by a low, smooth hill. SGD, as well as as a standard Langevin process, may jump between the two minima, while the average of the SGD parameters may correspond to the hill in between!

[7] The following theorem ([44]), that says that the set of full rank matrices in $\mathbb{R}^{m \times n}$ is connected, may be relevant, since it implies that matrices of the same rank are on a smooth manifold while matrices of a different rank are on disconnected manifolds.

Theorem 3 *The set of all $m \times n$ rectangular real matrices of rank r has only one connected component when $m \neq n$ or $r < m = n$. Furthermore, all these connected components are connected by analytic regular arcs.*

Furthermore, the second order statistics of the last layer features are computed as:

$$\Sigma_W = \frac{1}{C}\sum_{c=1}^{C}\frac{1}{N}\sum_{n=1}(h(x_{cn})-\mu_c)(h(x_{cn})-\mu_c)^\top$$
$$\Sigma_B = \frac{1}{C}\sum_{c=1}^{C}(\mu_c-\mu_G)(\mu_c-\mu_G)^\top \tag{17}$$
$$\Sigma_T = \frac{1}{NC}\sum_{c=1,n=1}^{C,N}(h(x_{cn})-\mu_G)(h(x_{cn})-\mu_G)^\top.$$

Here, Σ_W measures the within-class-covariance of the features, Σ_B is the between-class-covariance, and Σ_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) Variability collapse states that the variance of the feature embeddings of samples from the same class tends to zero, or formally, $\text{Tr}(\Sigma_W) \to 0$.

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 $\left|\left| \frac{W^\top}{\|W\|_F} - \frac{M}{\|M\|_F} \right|\right| \to 0$, 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 $\text{argmax}_c \langle W_L^c, h(x) \rangle \to \text{argmin}_c \|h(x) - \mu_c\|_2$.

6.1 Binary Classification

We first consider the case of binary classification since this closely follows our development in the previous section. The loss function is the same one defined in 1, and we consider minimization using Stochastic Gradient Descent with a batch size of 1.
We can obtain the conditions for Neural Collapse from the convergence of SGD if we make the following assumption:

Assumption 1 (Symmetric Quasi-interpolation (Binary Classification)) *Consider a binary classification problem with inputs in a feature space $\mathcal{X}$ and labels space $\{+1, -1\}$. A classifier $f_W : \mathcal{X} \to \mathbb{R}$ symmetrically quasi-interpolates a training dataset $\mathcal{S} = \{(x_n, y_n)\}_{n=1}^N$ if for all training examples $f_{W\,n} = y_n f_W(x_n) = 1 - \epsilon$, where ϵ is the interpolation gap.*

Though we do not carry a formal derivation here, we believe that the assumption follows from our analysis of the gradient flow equation by the following argument. The margins f_n are all the same for $\lambda = 0$ because we assume zero loss, that is interpolation of all x_n. For $\lambda > 0$ the gap to interpolation is given by Equation 8, that is $\epsilon = \frac{\mu^2}{\mu^2 + \sigma^2 + \lambda}$, with $\sigma = 0$ for $\lambda = 0$. Assuming continuous dependence of σ on λ (and continuous dependence of the solution of the ODE equations 2) we expect the margins f_n to be very close to each other ($f_j - f_i << \epsilon, \quad \forall i, j$) with a difference that goes to zero for $\lambda \to 0$.
The assumption has also empirical support: we show in Figure 6 that all the margins converge to be roughly equal. Once within class variability disappears, and for all training samples, the last layer features collapse to their mean, then, the outputs and margins also collapse to the same value. We can see this in the left plot of Figure 5 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 Figure 6 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. Notice that our analysis of the origin of the SGD noise shows that "strict" NC1 never happens with SGD, in the sense that the margins are never, not even asymptotically, exactly equal to each other, but just very close!

Theorem 4 *Let $\mathcal{S} = \{(x_n, y_n)\}_{n=1}^N$ be a dataset. Let (ρ, V) be the parameters of a ReLU network f such that V_L has converged when training using SGD with batches of size 1 on the square loss with LN+WD. Let $\mu_+ = \frac{1}{N}\sum_{n=1, y_n=1}^N h(x_n)$, $\mu_1 = \frac{1}{N}\sum_{n=1, y_n=-1}^N h(x_n)$. Assume that f satisfies Assumption 1. Then, it also satisfies the conditions of Neural Collapse as described below.*

- *NC1: $\mu_+ = h(x_n)$ for all $n \in [N], y_n = 1$, $\mu_- = h(x_n)$ for all $n \in [N], y_n = -1$;*
- *NC2: $\mu_+ = -\mu_-$, which is the structure of an ETF with two vectors;*
- *NC3: $V_L \propto \mu_+, \mu_-$;*
- *NC4: $sign(\rho f_V(x)) = \arg\min_{c \in \{+1,-1\}} \|\mu_c - h(x)\|$.*

Proof The update equations for SGD on the square loss function with LN+WD are given by:

$$\begin{aligned} V_L^{(t+1)} &= V_L^{(t)} - \eta \frac{\partial \mathcal{L}}{\partial V_L^{(t)}} \\ \implies V_L^{(t+1)} &= V_L^{(t)} - \eta \times \left(2\rho(\rho \bar{f}_n - 1) y_n h(x_n) + 2\nu_L^{(t)} V_L^{(t)}\right) \end{aligned} \tag{18}$$

We can apply the unit norm constraints $||V_L^{(t+1)}||^2 = 1$ and $||V_L^{(t)}||^2 = 1$ and ignore all terms that are $O\left(\eta^2\right)$ to compute $\nu_L^{(t)}$ as:

$$\begin{aligned} 2\nu_L^{(t)} &= 2\rho y_n V_L^{(t)\top} h(x_n)(1 - \rho \bar{f}_n) \\ \implies \nu_L^{(t)} &= \rho \bar{f}_n (1 - \rho \bar{f}_n) \end{aligned} \tag{19}$$

This gives us the following SGD update:

$$V_L^{(t+1)} = V_L^{(t)} - \eta \times 2\rho y_n (\rho \bar{f}_n - 1) \left(h(x_n) - f_n V_L^{(t)}\right) \tag{20}$$

Using assumption 1, we can see that for every training sample in class $y_n = 1$, $h(x_n) = \frac{(1-\epsilon)}{\rho} V_L$, and for every training sample in class $y_n = -1$, $h(x_n) = \frac{(-1+\epsilon)}{\rho} V_L$. This shows that within class variability has collapsed and that all last layer features collapse to their mean, which is the condition for NC1. We can also see that $\mu_+ = -\mu_-$, which is the condition for NC2 when there are 2 vectors in the Simplex ETF. The SGD convergence condition also tells us that $V_L \propto \mu_+$ and $V_L \propto \mu_-$, which gives us the NC3 condition. NC4 follows then from NC1-NC2, as shown by theorems in [11] ■

We can also obtain the following relationship between the ϵ in the margin and the weight decay parameter λ

Lemma 7 *For binary classification trained under the square loss, convergence of SGD with LN+WD to points satisfying assumption 1 (that is $\sigma = 0$) results in a margin of $1 - \epsilon$ with $\epsilon = \frac{\lambda}{\mu^2 + \lambda}$.*

Proof We consider a fixed network such that $\rho\mu = 1$ when $\lambda = 0$, that is the network interpolates the data. We now consider the smaller ρ_λ, associated with $\lambda = 0$ given by Equation 7, and compute $\epsilon = 1 - \rho_\lambda \mu = 1 - \frac{\mu^2}{\mu^2 + \lambda} = \frac{\lambda}{\mu^2 + \lambda}$. ■

6.2 Multiclass Classification

In the rest of this section we consider the phenomenon of Neural Collapse under the square loss with Weight Decay for the multiclass case. Because of the multiclass framework we cannot use our previous analysis of the dynamics of SGD and we have to make a specific assumption on the margins of quasi-interpolation for the different classes.

We consider a classification problem with C classes with a balanced training dataset $\mathcal{S} = \cup_{c=1}^{C} \mathcal{S}_c = \cup_{c=1}^{C} \{(x_{cn}, c)\}_{n=1}^{N} = \{(x_n, y_n)\}$ that has N training examples $\mathcal{S}_c = \{(x_{cn}, c)\}_{n=1}^{N}$ per-class $c \in [C]$. The labels are represented by the unit vectors $\{e_c\}_{c=1}^{C}$ in $\mathbb{R}^C$. We consider a deep ReLU network $f_W : \mathbb{R}^d \to \mathbb{R}^C$, which takes the form $f_W(x) = W_L \sigma(W_{L-1} \dots W_2 \sigma(W_1 x) \dots)$. However, we stick to the normalized reparameterization of the deep ReLU network as $f(x) = \rho V_L \sigma(V_{L-1} \dots V_2 \sigma(V_1 x) \dots)$. We train this normalized network with Stochastic Gradient Descent on the square loss with Lagrange Multipliers and Weight Decay. This architecture differs from the one considered in section 4 in that it has C outputs instead of a scalar output. Let the output of the network be $\rho f_V(x) = [\rho f_V^{(1)}(x) \dots \rho f_V^{(C)}(x)]^\top$, and the one-hot target vectors be $y_n = [y_n^{(1)} \dots y_n^{(C)}]^\top$. We will also follow the notation of [11] and use $h : \mathbb{R}^d \to \mathbb{R}^p$ to denote the last layer features of the deep network. This means that $f_V^{(c)}(x) = \langle V_L^c, h(x) \rangle$. Formally, the loss function with Lagrange Multipliers and Weight Decay takes the form:

$$\mathcal{L}_{\mathcal{S}}(\rho, \{V_k\}_{k=1}^{L}) = \frac{1}{NC} \sum_{c=1}^{C} \sum_{n=1}^{N} ||y_{cn} - \rho f_V(x_{cn})||^2 + \sum_{k=1}^{L} \nu_k \left(||V_k||^2 - 1 \right) + \lambda \rho^2 \tag{21}$$

At each time point t the optimization process selects a random class-balanced batch $\mathcal{S}' = \cup_{c=1}^{C} \cup_{n=1}^{b} \mathcal{S}_c'$ including B samples per-class from $\mathcal{S}_c' \subset \mathcal{S}_c$ and updates the scale and weights of the network is the following manner $V \leftarrow V - \eta \frac{\partial \mathcal{L}_{\mathcal{S}'}(\rho, V)}{\partial V}$, $\rho \leftarrow \rho - \eta \frac{\partial \mathcal{L}_{\mathcal{S}'}(\rho, V)}{\partial \rho}$ where $\eta > 0$ is a predefined learning rate and b is a divisor of N. A convergence point of the optimization process is a point (ρ, V) that will never be updated by any possible sequence of steps taken by the optimization algorithm. Specifically, the convergence points of the proposed method are all points ρ, V for which $\frac{\partial \mathcal{L}_{\mathcal{S}'}(\rho, V)}{\partial V} = 0$ and $\frac{\partial \mathcal{L}_{\mathcal{S}'}(\rho, V)}{\partial \rho} = 0$ for all class-balanced batches $\mathcal{S}' \subset \mathcal{S}$.

We make now the key assumption for the multiclass case[8]. The assumption is that the solution obtained by Stochastic Gradient Descent satisfies the follwing condition:

Assumption 2 (Symmetric Quasi-interpolation (Multiclass Classification)) *Consider a C-class classification problem with inputs in a feature space $\mathcal{X}$ and labels space $\mathbb{R}^C$. A classifier $f : \mathcal{X} \to \mathbb{R}^C$ symmetrically quasi-interpolates a training dataset $\mathcal{S} = \cup_{c=1}^{C} \mathcal{S}_c = \cup_{c=1}^{C} \{(x_{cn}, y_{cn})\}_{n=1}^{N}$ if for all training examples x_{cn} in class c, $f^{(c)}(x_{cn}) = 1 - \epsilon$, and $f^{(c')}(x_{cn}) = \frac{\epsilon}{C-1}$, where ϵ is the interpolation gap.*

The main result of this section is

Theorem 5 *Let $\mathcal{S} = \cup_{c=1}^{C} \{(x_{cn}, c)\}_{n=1}^{N}$ be a dataset and B be a divisor of N. Let (ρ, V) be the parameters of a ReLU network f_W such that V_L has converged when training using SGD with balanced batches of size $B = bC$ on the square loss with LN+WD. Let $\mu_c = \frac{1}{N} \sum_{n=1}^{N} h(x_{cn})$, $\mu_G = \frac{1}{C} \sum_{c=1}^{C} \mu_c$ and $M = [\dots \mu_c - \mu_G \dots] \in \mathbb{R}^{p \times C}$. Assume that f_W satisfies Assumption 2. Then, it also satisfies the conditions of Neural Collapse as described below.*

- *NC1: $\mu_c = h(x_{cn})$ for all $n \in [N]$;*
- *NC2: the vectors $\{\mu_c - \mu_G\}_{c=1}^{C}$ form an ETF;*
- *NC3: $V_L^\top = \frac{M}{\|M\|_F}$;*
- *NC4: $\arg\max_{c \in [C]} f_W^c(x) = \arg\min_{c \in [C]} \|\mu_c - h(x)\|$.*

Proof Our training objective is the loss function described in (21). The network is trained using SGD along with Lagrange normalization and weight decay. We use SGD with balanced batches to train the network. Each step taken by SGD takes the form $-\eta \frac{\partial \mathcal{L}_{\mathcal{S}'}}{\partial V}$, where $\mathcal{S}' \subset \mathcal{S}$ is a balanced batch containing exactly b samples per class. We consider limit points of the learning procedure, meaning that $\frac{\partial \mathcal{L}_{\mathcal{S}'}}{\partial V} = 0$

[8] We conjecture it should follow from a multiclass analysis of the dynamics.

for all balanced batches $\mathcal{S}'$. Let $\mathcal{S}' = \cup_{c=1}^{C} \cup_{n=1}^{b} \{(\hat{x}_{cn}, \hat{y}_{cn})\}$ be such a balanced batch. We use Stochastic Gradient Descent, where at each time t the batch $\mathcal{S}'$ is drawn at random from $\mathcal{S}$, to study the time evolution of the normalized parameters V_L in the limit $\eta \to 0$.

$$\begin{aligned} V_L^{(t+1)} &= V_L^{(t)} - \eta \frac{\partial \mathcal{L}_{\mathcal{S}'}}{\partial V_L^{(t)}} \\ \implies V_L^{(t+1)} &= V_L^{(t)} - \eta \times \left(\frac{1}{B} \sum_{c'=1}^{C} \sum_{n=1}^{b} 2\rho(\rho f_V(x_{c'n}) - y_{c'n}) h(x_{c'n})^\top + 2\nu_L^{(t)} V_L^{(t)} \right) \end{aligned} \tag{22}$$

We can apply the unit norm constraints $||V_L^{(t+1)}||_F^2 = \text{tr}(V_L^{(t+1)\top} V_L^{(t+1)}) = 1$ and $||V_L^{(t)}||_F^2 = \text{tr}(V_L^{(t)\top} V_L^{(t)}) = 1$ and ignore all terms that are $O\left(\eta^2\right)$ to compute $\nu_L^{(t)}$ as:

$$\begin{aligned} 2\nu_L^{(t)} &= -\frac{1}{B} \sum_{c'=1}^{C} \sum_{n=1}^{b} 2\rho \text{tr}\left(V_L^{(t)\top} (\rho f_V(x_{c'n}) - y_{c'n}) h(x_{c'n})^\top \right) \\ \implies \nu_L^{(t)} &= -\frac{1}{B} \sum_{c'=1}^{C} \sum_{n=1}^{b} \rho \text{tr}\left((V_L^{(t)} h(x_{c'n}))^\top (\rho f_V(x_{c'n}) - y_{c'n}) \right) \\ &= -\frac{1}{B} \sum_{c'=1}^{C} \sum_{n=1}^{b} \rho f_V(x_{c'n})^\top (\rho f_V(x_{c'n}) - y_{c'n}) \end{aligned} \tag{23}$$

This means that the (stochastic) gradient of the loss with respect to the last layer V_L, and each classifier vector V_L^c with Lagrange Normalization can be written as (we drop the time index t for clarity):

$$\begin{aligned} \frac{\partial \mathcal{L}_{\mathcal{S}'}}{\partial V_L} &= \frac{-2\rho}{B} \sum_{c'=1}^{C} \sum_{n=1}^{b} f_V(x_{c'n})^\top (\rho f_V(x_{c'n}) - y_{c'n}) V_L - (\rho f_V(x_{c'n}) - y_{c'n}) h(x_{c'n})^\top \\ \frac{\partial \mathcal{L}_{\mathcal{S}'}}{\partial V_L^c} &= \frac{-2\rho}{B} \sum_{c'=1}^{C} \sum_{n=1}^{b} f_V(x_{c'n})^\top (\rho f_V(x_{c'n}) - y_{c'n}) V_L^c - (\rho f_V^{(c)}(x_{c'n}) - y_{c'n}^{(c)}) h(x_{c'n}) \end{aligned} \tag{24}$$

Let us analyze the equilibrium parameters at the last layer, considering each classifier vector V_L^c of V_L, separately:

$$\begin{aligned} 0 = \frac{\partial \mathcal{L}_{\mathcal{S}'}}{\partial V_L^c} &= \frac{2\rho}{B} \sum_{c'=1}^{C} \sum_{n=1}^{b} f_V(x_{c'n})^\top (\rho f_V(x_{c'n}) - y_{c'n}) V_L^c - (\rho f_V^{(c)}(x_{c'n}) - y_{c'n}^{(c)}) h(x_{c'n}) \\ &= \frac{2\rho}{B} \sum_{n=1}^{b} f_V(x_{cn})^\top (\rho f_V(x_{cn}) - y_{cn}) V_L^c - (\rho f_V^{(c)}(x_{cn}) - 1) h(x_{cn}) \\ &\quad + \frac{2\rho}{B} \sum_{c' \in [C] \setminus \{c\}} \sum_{n=1}^{b} f_V(x_{c'n})^\top (\rho f_V(x_{c'n}) - y_{c'n}) V_L^c - \rho f_V^{(c)}(x_{c'n}) h(x_{c'n}) \end{aligned} \tag{25}$$

Let us consider solutions that achieve *symmetric quasi-interpolation*, with $\rho f_V^{(c)}(x_{cn}) = 1 - \epsilon$, and $\rho f_V^{(c)}(x_{c'n}) = \frac{\epsilon}{C-1}$. Hence, we have

$$\frac{2\rho}{B} \sum_{n=1}^{b} \epsilon h(x_{cn}) - \frac{2\rho}{B} \sum_{c' \in [C] \setminus \{c\}} \sum_{n=1}^{b} \frac{\epsilon}{C-1} h(x_{c'n}) - 2\left(\epsilon(1-\epsilon) - \frac{\epsilon^2}{C-1} \right) V_L^c = 0. \tag{26}$$

In addition, consider a second batch $\mathcal{S}''$ that differs from $\mathcal{S}'$ by only one sample x'_{cn} instead of x_{cn} from class c. By applying the previous Eq. (26) for $\mathcal{S}'$ and for $\mathcal{S}''$, we can obtain $h(x_{cn}) = h(x'_{cn})$, which proves NC1.

Let $\mathcal{S} = \cup_{i=1}^{k} \mathcal{S}^i$ be a partition of $\mathcal{S}$ into $k = N/b$ (an integer) disjoint batches. Since our data is balanced, we obtain that

$$\begin{aligned} 0 &= \frac{1}{k}\sum_{i=1}^{k} \frac{\partial \mathcal{L}_{\mathcal{S}^i}(\rho, V)}{\partial V_L^c} \\ &= \frac{\partial \mathcal{L}_{\mathcal{S}}(\rho, V)}{\partial V_L^c} \\ &= \frac{2\rho}{NC}\sum_{c'=1}^{C}\sum_{n=1}^{N} f_V(x_{c'n})^\top(\rho f_V(x_{c'n}) - y_{c'n})V_L^c - (\rho f_V^{(c)}(x_{c'n}) - y_{c'n}^{(c)})h(x_{c'n}) \\ &= \frac{2\rho}{NC}\sum_{n=1}^{N}\epsilon h(x_{cn}) - \frac{2\rho}{NC}\sum_{c'\in[C]\setminus\{c\}}\sum_{n=1}^{N}\frac{\epsilon}{C-1}h(x_{c'n}) - 2\left(\epsilon(1-\epsilon) - \frac{\epsilon^2}{C-1}\right)V_L^c \end{aligned} \tag{27}$$

Under the conditions of NC1 we can simply write $\mu_c = h(x_{cn})$ for all $n \in [N]$ and $c \in [C]$. Let us denote the global feature mean by $\mu_G = \frac{1}{C}\sum_{c=1}^{C}\mu_c$. This means we have:

$$\frac{\partial \mathcal{L}_{\mathcal{S}}(\rho, V)}{\partial V_L^c} = 0 \quad \implies V_L^c = \frac{\rho}{((C-1)(1-\epsilon)-\epsilon)} \cdot (\mu_c - \mu_G). \tag{28}$$

This implies that the last layer parameters V_L are a scaled version of the centered class-wise feature matrix $M = [\dots \mu_c - \mu_G \dots]$. Thus at equilibrium, with quasi interpolation of the training labels, we obtain $\frac{V_L^\top}{\|V_L\|_F} = \frac{M}{\|M|_F}$.
From the SGD 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:

$$\begin{aligned} \|V_L^c\|_2^2 &= \frac{\frac{1}{NC}\sum_{c'=1}^{C}\sum_{n=1}^{N}(\rho f_V^{(c)}(x_{c'n}) - y_{c'n}^{(c)})\rho f_V^{(c)}(x_{c'n})}{\frac{1}{NC}\sum_{c'=1}^{C}\sum_{n=1}^{N}\langle \rho f_V(x_{c'n}) - y_{c'n}, \rho f_V(x_{c'n})\rangle} \\ &= \frac{\frac{\epsilon}{C}(1-\epsilon) - \frac{\epsilon^2}{C(C-1)}}{\epsilon(1-\epsilon) - \frac{\epsilon^2}{C-1}} = \frac{1}{C} \end{aligned} \tag{29}$$

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 Equation (28)

$$\begin{aligned} \langle V_L^c, \mu_G\rangle &= \frac{1-\epsilon}{\rho} - \frac{\left(1 - \frac{C}{C-1}\epsilon\right)(C-1)}{\rho}\|V_L^c\|_2^2 \\ &= \frac{1-\epsilon}{\rho} - \frac{\frac{C-1}{C} - \epsilon}{\rho} = \frac{1}{\rho C}. \end{aligned} \tag{30}$$

From the quasi-interpolation of the incorrect class labels, we have that $\langle V_L^c, \mu_{c'}\rangle = \frac{\epsilon}{\rho(C-1)}$, which means $\langle V_L^c, \mu_{c'} - \mu_G\rangle + \langle V_L^c, \mu_G\rangle = \frac{\epsilon}{\rho(C-1)}$. Plugging in the previous result and using (29) yields

$$\begin{aligned} \frac{(C-1)\left(1 - \frac{C}{C-1}\epsilon\right)}{\rho} \times \langle V_L^c, V_L^{c'}\rangle &= \frac{\epsilon}{\rho(C-1)} - \frac{1}{\rho C} \\ \implies \langle \tilde{V}_L^c, \tilde{V}_L^{c'}\rangle &= \frac{1}{\|V_L^c\|_2^2} \times \frac{-1}{C(C-1)} = -\frac{1}{C-1} \end{aligned} \tag{31}$$

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 [11]. ■

Remarks

- In a homogeneous neural network, the absence of interpolation is a necessary condition for Neural Collapse. We can see this by plugging in the Neural Collapse solution into the square loss function and observing that the loss is non-zero, which means the Neural Collapse solution is not a global minimum. This means that weight decay regularization is necessary to observe Neural Collapse for homogeneous neural networks.
- The analysis of the loss landscape and of the qualitative dynamics under the square loss in section 4.3 and in section 4.2 implies that all quasi-interpolating solutions with $\rho \geq \rho_0$ and $\lambda > 0$ and satisfying assumption 2 yield neural collapse and have its four properties.
- SGD is a necessary requirement in our proof of NC1.
- From our analysis, the relationship between Neural Collapse and generalization is not evident. We believe that NC1 to NC4 should take place for any quasi-interpolating solutions (in the square loss case), including solutions that do not have a large margin (they have small ρ). In particular the analysis above predicts Neural Collapse for datasets with significant amounts of label noise, or even random labels, .

7 Generalization, Rademacher complexity and norms

8 Generalization, Rademacher complexity and norms

We begin with basic generalization bounds that hold with probability at least $(1-\delta)$, $\forall g \in \mathbb{G}$ of the form [45]:

$$|L(g) - \hat{L}(g)| \leq 2\mathbb{R}_N(\mathbb{G}) + \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}} \tag{32}$$

where $L(g) = \mathbf{E}[\ell_\gamma(g(x), y)]$ is the expected loss, $\hat{L}(g)$ is the empirical loss, $\mathbb{R}_N(\mathbb{G})$ is the empirical Rademacher average of the class of functions $\mathbb{G}$ measuring its complexity. In our case L is the square loss.
Assume that the square loss on the training set is exactly zero (which usually almost is) and the margins $\bar{f}_n$ are all the same and very close to 1 (these assumptions can be weakened). Then we can measure the loss of the functions obtained via square loss minimization using the *ramp loss*[9]. Under our assumption of $g_n(x_n)y_n = 1 \quad \forall n$ we can replace the square loss with $\ell_\gamma(g(x), y)$ with $gamma$ very close to 1: the ramp loss will effectively measure the classification error for the training set. Notice that it is possible to associate to g an indicator function $\frac{1-y_i g(x_i}{2}$ (see [46]) and that then

$$err(g) - \widehat{err(g)} \leq \mathbb{R}_N(\mathbb{G}) + \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}} \tag{33}$$

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

$$\mathbb{R}_N(\mathbb{G}) = \rho \mathbb{R}_N(\mathbb{F}), \tag{34}$$

where $\mathbb{G}$ is the space of functions of our unnormalized networks and $\mathbb{F}$ denotes the corresponding normalized networks. This yields [47]

[9] The ramp loss is defined as

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

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

$$|err(g) - \widehat{err(g)}| \leq \rho \mathbb{R}_N(\mathbb{F}) + \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}}, \tag{35}$$

implying the following bound for interpolating minimizers

$$err(g) \leq \rho \mathbb{R}(\mathbb{F}) + \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}}. \tag{36}$$

Furthermore we can bound $\mathbb{R}(\mathbb{F})$ (see Equation 5 in [48]) as

$$\mathbb{R}_N(\mathbb{F}) \leq \frac{\sqrt{L}}{\sqrt{N}}. \tag{37}$$

Thus the bound on $\mathbb{R}_N(\mathbb{F})$ depends on the architecture of the network but not on the training set. It could be improved further if the spectral norm of the weight matrices is close to their Frobenius norm (see Equation 6 in [48]), which happens with small rank of the weight matrices. The overall bound is then (assuming zero training error)

$$err(g) \leq \frac{\rho}{\sqrt{N}}\sqrt{L} + \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}}. \tag{38}$$

Relative Generalization 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 ρ. Let us call them $g^a(x) = \rho_a f^a(x)$ and $g^b(x) = \rho_b f^b(x)$. Using the notation of this paper, the functions f_a and f_b correspond to networks with normalized weight matrices at each layer.
Let us assume that $\rho_a < \rho_b$.
We now use use Equation 35 and the fact that the empirical $\hat{L_\gamma}$ *for both functions is the same* to write $L_0(f^a) = L_0(F^a) \leq c_1 \rho_a \mathbb{R}_N(\tilde{\mathbb{F}}) + c_2\sqrt{\frac{\ln(\frac{1}{\delta})}{2N}}$ and $L_0(f^b) = L_0(F^b) \leq c_1 \rho_b \mathbb{R}_N(\tilde{\mathbb{F}}) + c_2\sqrt{\frac{\ln(\frac{1}{\delta})}{2N}}$. The bounds have the form

$$L_0(f^a) \leq A\rho_a + \epsilon \tag{39}$$

and

$$L_0(f^b) \leq A\rho_b + \epsilon \tag{40}$$

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. As a consequence this result does not guarantee that a solution with smaller ρ will always have a smaller expected error than a solution with larger ρ.
Notice that the this generalization claim is just a *relative* claim about different solutions obtained with the same network trained on the same training set.
Figure 8 shows clearly that increasing the percentage of random labels increases ρ needed to maintain interpolation – decreasing the margin – and that at the same time the test error increases as expected from Equation 35. This monotonic relation between margin and accuracy at test seems to break down for small differences in margin as shown in Figure 9, though the significance of the effect is unclear. Of course this kind of behavior is not inconsistent with an upper bound.

The bound ρ is surprisingly small As we mentioned earlier, we bound the Rademacher complexity of the convolutional layers not by the norm of the associated Toeplitz matrices but by *the norm of the filters*. The reason is that the covering numbers associated with convolution are much smaller because of shift invariance (see section 3.3 in [37] and theorems in [38]). The ρ_k calculated in this way for the convolutional layers of our network reduces the generalization bounds by several orders of magnitude, despite still being vacuous (see Figure 10). It is thus many orders of magnitude better than VC bounds which depend on the number of weights and are therefore always vacuous in overparametrized situations. With norm-based bounds it is possible to have overparametrization and interpolation simultaneously with non-vacuous generalization bounds. In fact a slightly different 5-layers network with a single last non-convolutional layer achieves a non-vacuous norm-based bound at around $20,000$ training data which is much less than the total number of parameters $(246,886)$.

We remark that it is impossible to compare directly experiments with ρ obtained with $\lambda = 0$ and with ρ obtained with $\lambda > 0$. The reason is that in the case of $\lambda = 0$ we can expect to find solutions to $\rho\overline{f_n} = 1$, for most values of ρ, given sufficient overparametrization. Without regularization (and especially if normalization is used) the solution will be found close to the ρ associated with initialization. In this situation, similar to linear regression of random features, it seems likely that optimization mainly acts on the weights of the last layer. Alternatively, this is a case in which the bound should be better estimated in terms of the norm $||W_k - A_k||$ where the A_k are appropriate reference matrices [49]: here the A_k could be chosen to be the weight matrices at initialization instead of zero matrices[10].
Different bounds [49] focused on the spectral norm of the weight matrices could be used

$$\mathbb{R}_N(\mathbb{F}) \leq \frac{\Pi_{j=1}^{L}||V_j||_s L^{\frac{3}{2}}}{\sqrt{N}}, \tag{41}$$

where $||V_j||_s$ are the spectral norms of the matrices V_j that have Frobenius norm 1. In a future paper, we plan to compare these different bounds in experiments using BN and LN with the goal of checking which bounds may not be vacuous and why BN is usually better than LN.

9 Summary and Discussion

In this paper we have considered a specific model of the dynamics of, first, gradient flow (GF), and then SGD, in overparametrized ReLU neural networks trained for square loss minimization. Under the assumption of convergence to zero loss minima, we have shown that solutions have a bias toward small ρ, defined as the product of the Frobenius norms of each layer's (unnormalized) weight matrix. We assume that during training there is normalization using a Lagrange multiplier (LN) of each layer weight matrix but the last one, together with Weight Decay (WD) with the regularization parameter λ. Without weight decay, the solution would be the interpolating solution with minimum ρ. In the absence of LN and WD, good solutions for classification may still be achieved because of the implicit bias towards small norm solutions in the GD dynamics introduced by carefully chosen close-to-zero initial conditions on the norms of the weights. However, for $\lambda = 0$ we often observe solutions with large ρ that are suboptimal and probably similar to the NTK regime.
A puzzle that remains open is why BN leads to better solutions than LN and WN, despite the many similarities between them. WN is easier to formalize mathematically as LN, which is the main reason for the role it plays in this paper.
With SGD and weight decay we uncover a bias of convergence towards not only small ρ but also towards small rank solutions. At the same time, we show that SGD with regularization yields an unavoidable noise, which makes exact convergence impossible, even asymptotically. However, margins do converge to each other within a small ϵ, implying that the first condition for neural collapse [11] is satisfied in this approximate sense. We show that the other conditions are also satisfied.
A natural question is whether Neural Collapse is related to solutions with good generalization. Our analysis suggests that this is not the case, at least not directly: Neural Collapse is a property of the dynamics, independently of the size of the margin which provides an upper bound on the expected error[11]. In fact, our prediction of Neural Collapse for randomly labeled CIFAR10, was confirmed originally in preliminary experiments by our collaborators (Papyan et al.) and more recently in other papers (see for instance, [27]).
Independently of Neural Collapse, can our analysis of the square loss dynamics and its connection with margin and low rank provide insights on generalization of the solutions of gradient descent techniques? Large margin is usually associated with good generalization [45]; in the meantime, however, it is also broadly recognized that margin alone does not fully account for generalization in deep nets [50, 51, 52]. Margin in fact provides an upper bound on generalization error, as shown in section 8. Larger margin gives a better upper bound on the generalization error for the same network trained on the same data. We have verified empirically this property by varying the margin using different degrees of random labels in a binary classification task. While training gives perfect classification and zero square loss, the margin on the training set together with the test error decreases with the increase in the percentage of random labels. The upper bound given in section 8, however, does not explain by itself details of

[10] In our experiments with $\lambda > 0$, see Figure **??**, $||W_k - A_k|| \approx ||W_k||$
[11] Even if margin is likely to be just one of the factors determining out-of-sample performance.

the generalization behavior that we observe for different initializations (see Figure 9), where small differences in margin are actually anticorrelated with small differences in test error. We conjecture that margin together with rank may be sufficient to explain generalization, as discussed briefly in section 8. The bias towards small ρ solutions induced by regularization for $\lambda > 0$ can be replaced by an implicit bias induced by small initialization and parameters values that allow convergence to the first quasi-interpolating solution for increasing ρ. The main effect of $\lambda > 0$ is to eliminate degeneracy of the dynamics at the zero-loss critical points.

For $\lambda > 0$ a main property of the minimizers that upper bounds their expected error is ρ, which is the inverse of the margin: we prove that among all the quasi-interpolating solutions the ones associated with smaller ρ have better bounds on the expected classification error. Our norm-based bounds under LN training are vacuous in our experiments but within a couple of orders of magnitude of the expected error. They are thus better than previous bounds, such as VC bounds, and better than argued in [53], mainly because the covering numbers associated with the convolutional layers are estimated by taking into account the associated invariance properties. This is encouraging since in our experiments there is large overparametrization but still relatively low complexity.

The situation here is somewhat similar to the linear case: for overparametrized networks the best solution in terms of generalization is the minimum norm solution towards which GD is biased. Associated with the minimum ρ solutions are upper bounds on the generalization error and properties such as Neural Collapse. Interestingly, these results do not say why deep networks should be better than other classifiers such as kernel machines. Our analysis supports the idea that the advantage of deep networks relative to other standard classifiers is greater for the problems to which specific deep architectures such as CNNs can be applied. The deep reason is that CNNs reflect the function graph of certain locally compositional target function – which have small intrinsic dimensionality – and thus can be approximated well by sparse networks without incurring in the curse of dimensionality. Despite overparametrization the sparse networks can show good generalization (because of the "small" covering numbers).

In summary, we describe how gradient descent on the square loss in the presence of regularization can converge to minimum ρ solutions, corresponding to max margin solutions, and show that they are biased towards small rank of the weight matrices. Associated with these solutions are properties such as Neural Collapse and bounds on the generalization error. Our analysis of the square loss suggests a bias of SGD in the presence of weight decay towards minimum ρ and low rank solutions.

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

[1] Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. *CoRR*, abs/1906.05890, 2019.

[2] Tomaso Poggio, Andrzej Banburski, and Qianli Liao. Theoretical issues in deep networks. *PNAS*, 2020.

[3] Mor Shpigel Nacson, Suriya Gunasekar, Jason D. Lee, Nathan Srebro, and Daniel Soudry. Lexicographic and Depth-Sensitive Margins in Homogeneous and Non-Homogeneous Deep Models. *arXiv e-prints*, page arXiv:1905.07325, May 2019.

[4] A. Banburski, Q. Liao, B. Miranda, T. Poggio, L. Rosasco, B. Liang, and J. Hidary. Theory of deep learning III: Dynamics and generalization in deep networks. *CBMM Memo No. 090*, 2019.

[5] Like Hui and Mikhail Belkin. Evaluation of neural architectures trained with square loss vs cross-entropy in classification tasks. *arXiv preprint arXiv:2006.07322*, 2020.

[6] Ryan M. Rifkin. *Everything Old Is New Again: A Fresh Look at Historical Approaches to Machine Learning*. PhD thesis, Massachusetts Institute of Technology, 2002.

[7] T. Poggio and Q. Liao. Generalization in deep network classifiers trained with the square loss. *CBMM Memo No. 112*, 2019.

[8] T. Poggio and Y. Cooper. Loss landscape: Sgd has a better view. *CBMM Memo 107*, 2020.

[9] Yaim Cooper. Global minima of overparameterized neural networks. *SIAM Journal on Mathematics of Data Science*, 3(2):676–691, 2021.

[10] Nadav Timor, Gal Vardi, and Ohad Shamir. Implicit regularization towards rank minimization in relu networks. *CoRR*, abs/2201.12760, 2022.

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

[12] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

[13] Lenaic Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on Learning Theory*, pages 1305–1338. PMLR, 2020.

[14] Tengyu Xu, Yi Zhou, Kaiyi Ji, and Yingbin Liang. When will gradient methods converge to max-margin classifier under relu models? *Stat*, 10(1):e354, 2021.

[15] Vidya Muthukumar, Adhyyan Narang, Vignesh Subramanian, Mikhail Belkin, Daniel Hsu, and Anant Sahai. Classification vs regression in overparameterized regimes: Does the loss function matter? *arXiv e-prints*, page arXiv:2005.08054, May 2020.

[16] Tengyuan Liang and Alexander Rakhlin. Just Interpolate: Kernel "Ridgeless" Regression Can Generalize. *arXiv e-prints*, page arXiv:1808.00387, Aug 2018.

[17] Tengyuan Liang and Benjamin Recht. Interpolating classifiers make few mistakes. *arXiv preprint arXiv:2101.11815*, 2021.

[18] Kai Zhong, Zhao Song, Prateek Jain, Peter L Bartlett, and Inderjit S Dhillon. Recovery guarantees for one-hidden-layer neural networks. In *International conference on machine learning*, pages 4140–4149. PMLR, 2017.

[19] Mahdi Soltanolkotabi, Adel Javanmard, and Jason D Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 65(2):742–769, 2018.

[20] Simon S. Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019.

[21] Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. *arXiv preprint arXiv:1812.07956*, 2018.

[22] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *arXiv preprint arXiv:1806.07572*, 2018.

[23] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

[24] Zhengdao Chen, Grant M Rotskoff, Joan Bruna, and Eric Vanden-Eijnden. A dynamical central limit theorem for shallow neural networks. *arXiv preprint arXiv:2008.09623*, 2020.

[25] Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pages 322–332. PMLR, 2019.

[26] Dustin G. Mixon, Hans Parshall, and Jianzong Pi. Neural collapse with unconstrained features. *CoRR*, abs/2011.11619, 2020.

[27] Zhihui Zhu, Tianyu Ding, Jinxin Zhou, Xiao Li, Chong You, Jeremias Sulam, and Qing Qu. A geometric analysis of neural collapse with unconstrained features. *Advances in Neural Information Processing Systems*, 34:29820–29834, 2021.

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

[29] Cong Fang, Hangfeng He, Qi Long, and Weijie J. Su. Layer-peeled model: Toward understanding well-trained deep neural networks. *CoRR*, abs/2101.12699, 2021.

[30] Stephan Wojtowytsch et al. On the emergence of tetrahedral symmetry in the final and penultimate layers of neural network classifiers. *arXiv preprint arXiv:2012.05420*, 2020.

[31] Tolga Ergen and Mert Pilanci. Revealing the structure of deep neural networks via convex duality. *arXiv preprint arXiv:2002.09773*, 2020.

[32] T. Poggio and Q. Liao. Generalization in deep network classifiers trained with the square loss. *Center for Brains, Minds and Machines (CBMM) Memo No. 112*, 2021.

[33] XY Han, Vardan Papyan, and David L Donoho. Neural collapse under mse loss: Proximity to and dynamics on the central path. *arXiv preprint arXiv:2106.02073*, 2021.

[34] Jinxin Zhou, Xiao Li, Tianyu Ding, Chong You, Qing Qu, and Zhihui Zhu. On the optimization landscape of neural collapse under mse loss: Global optimality with unconstrained features. *arXiv preprint arXiv:2203.01238*, 2022.

[35] Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. *CoRR*, abs/1812.03981, 2018.

[36] Sourav Chatterjee. Convergence of gradient descent for deep neural networks, 2022.

[37] Fabio Anselmi, Lorenzo Rosasco, and Tomaso A. Poggio. On invariance and selectivity in representation learning. *CoRR*, abs/1503.05938, 2015.

[38] Antoine Ledent, Yunwen Lei, and Marius Kloft. Improved generalisation bounds for deep learning through l^∞ covering numbers. *CoRR*, abs/1905.12430, 2019.

[39] A Krizhevsky. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

[40] Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes. Fisher-rao metric, geometry, and complexity of neural networks. *CoRR*, abs/1711.01530, 2017.

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

[42] Quynh Nguyen. On connected sublevel sets in deep learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4790–4799. PMLR, 09–15 Jun 2019.

[43] T. Galanti and T. Poggio. Sgd noise and implicit low-rank bias in deep neural networks. *Center for Brains, Minds and Machines (CBMM) Memo No. 134*, 2022.

[44] J. C. Evard and F. Jafari. The set of all rectangular real matrices is connected by analytic regular arcs. *Proceedings of the American Mathematical Society*, 120(2):413–419, February 1994.

[45] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Summer school on machine learning*, pages 169–207. Springer, 2003.

[46] Rob Shapire. Cos 511: Theoretical machine learning, lecture 10. 2018.

[47] Maria-Florina Balcan. 10-806 foundations of machine learning and data science, lecture 8: October 5, 2015. 2015.

[48] Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. *CoRR*, abs/1712.06541, 2017.

[49] Peter L. Bartlett, Dylan J. Foster, and Matus Telgarsky. Spectrally-normalized margin bounds for neural networks. *CoRR*, abs/1706.08498, 2017.

[50] P. Bartlett, D. J. Foster, and M. Telgarsky. Spectrally-normalized margin bounds for neural networks. *ArXiv e-prints*, June 2017.

[51] Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. *arXiv preprint arXiv:1810.00113*, 2018.

[52] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. *arXiv preprint arXiv:1912.02178*, 2019.

[53] Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. *CoRR*, abs/1802.05296, 2018.

[54] David G. T. Barrett and Benoit Dherin. Implicit gradient regularization, 2021.

[55] Daniel Kunin, Javier Sagastuy-Brena, Surya Ganguli, Daniel L. K. Yamins, and Hidenori Tanaka. Neural mechanics: Symmetry and broken conservation laws in deep learning dynamics, 2021.
clearpage

A Critical points of SGD

Consider the case of $\lambda = 0$,

$$\min_W L(f(W)) = \min_W \sum_{i=1}^{N} \ell_i^2 \tag{42}$$

with $\ell_i = y_i - f(W; x_i)$.
We minimize $L(f(W))$ by running the following dynamical system (e.g., gradient flow)

$$\dot{W} = \nabla_W L(f(W)) = \sum_i^N \nabla_W f(W; x_i)(y_i - f(W; x_i)). \tag{43}$$

SGD can be formulated as follows. First define

Definition 6 *A random vector $v \in R^d$ drawn from a distribution $\mathcal{D}$ is a sampling vector if $\mathcal{E}_{\mathcal{D}}[v_i] = 1, \forall i$*

Then, the stochastic version of Equation (42) is

$$\min_W \mathcal{E}_{\mathcal{D}}[L(f(W))] = \min_W \mathcal{E}_{\mathcal{D}} \sum_i^n v_i \ell_i \tag{44}$$

Usually the distribution over $\mathcal{D}$ is assumed to be random v with independent components v_i, satisfying condition 6. The literature gives the impression that in expectation SGD is equal to GD, which is true *if* ℓ_i^2 are quadratic functions.
Finding the interpolating global minimizers of $L = \sum \ell_i^2$ is equivalent to finding the set of network weights W^* that solve the system of equations $\ell_i(W^*) = 0, \forall i = 1, \cdots, N$. Thus instead of finding all the critical points of the gradient of L, we would like to find the joint minimizers – that is the W – that minimize $\ell_i^2, \forall i = 1, \cdots, n$.
We define *critical points of SGD* the solutions of $\ell_i \nabla \ell_i = 0, \forall i$. For a discussion see [41].
A critical point of SGD with minibatch of size 1 is defined as $\dot{z} = g(x_n) = 0, \forall n = 1, \cdots, N$. This compares with a critical point of GD defined as $\dot{z} = \sum_{n=1}^{N} g(x_n) = 0$. GD of course is SGD with minibatch size N. What about SGD with intermediate minibatch sizes? The answer is[12]

Lemma 8 *If $\dot{z} = \sum_{n=1}^{M} g(x_n) = 0$ for enough random SGD draws of size $M < N$, then $\dot{z} = g(x_n) = 0, \forall n = 1, \cdots, N$.*

B Dynamics of ρ

Lemma 9
Assume $\rho \sum \overline{f_n} < 1$ and a normalized network, that is $\|V_k\| = 1, \quad k = 1, \cdots, L$. Then the loss can be written as $\mathcal{L}_{\mathcal{S}} = 1 - \rho(\frac{1}{2}\dot{\rho} + \mu)$.

Proof
Consider the loss $\mathcal{L}_{\mathcal{S}} = 1 - 2\rho\mu + \rho^2 M + \lambda\rho^2$ and $\dot{\rho} = 2\mu - 2\rho M - 2\lambda\rho$, which gives $2\rho M = 2\mu - 2\lambda\rho - \dot{\rho}$. Thus we obtain the result

$$\mathcal{L}_{\mathcal{S}} = 1 - \frac{1}{2}\rho\dot{\rho} - \rho\mu. \tag{45}$$

C Observation on ρ dynamics

- Immediately after initialization with $\rho(0) < \rho_0$, if there is convergence (depending on parameters values and network architecture, convergence is not guaranteed) ρ grows non-monotonically until $\rho(t)$ is within a neighborhood of ρ_{eq} (within $\lambda\rho_0 + \sum f_n^2$).

[12] A caution here is necessary: the number of random draws of size M from a data set of size N is enormous since it is equal to $\binom{N}{M}$ and thus, though all of them provide an over-constrained set of equations, only a very small subset of meaningful constraints may be available in practice.

- There may be oscillations in $\rho(t)$, that is time intervals in which $\dot{\rho} < 0$: these intervals are limited.
- $\dot{\rho} = 0$ between positive and negative values may be a critical point of the system if the $\dot{V}_k = 0$ for the associated ρ give compatible f values; these critical points may be local minima or saddle points; if local minima and saddles can be avoided (by SGD or by restarting optimization since $\mathcal{L}_S > 0$) the dynamics will eventually converge to an interpolation or quasi-interpolation solution with $\mathcal{L}_S = 0$ if $\lambda = 0$ and otherwise close to zero.
- Weight decay performs the traditional role of promoting solutions with small norm. In the case of large initialization, we can see from (7) that ρ_{eq} is determined by λ.
- Norm regularization is however not the only contribution of weight decay. The critical points $\dot{V}_k = 0$ may not be normalized properly – and thus may not impose a rank one constraint, see later – if the solution interpolates. In particular, an unnormalized interpolating solution can satisfy the equilibrium equations for $\dot{V}_k$. This is expected from the constrained dynamics which by itself constrains the norm of the V_k to not change during the iterations[13]. By preventing exact interpolation, weight decay ensures that the critical points of the V_k dynamics lie on a fixed Frobenius norm ball. As we will discuss later, by preventing exact interpolation, gradient flow with weight decay under the square loss shows the phenomenon of SGD noise and of Neural Collapse.
- In all these observations we have assumed gradient flow. The dynamics for gradient descent implies by itself the presence of damped oscillations in ρ. In addition, the randomness of SGD also contributes to transient decreases in the norm ρ.

D Margins for SGD

Consider first a global minimum with $\lambda = 0$ and $\mathcal{L}_S = 0$. Such a global minimum corresponds to interpolation of all data points and to $\dot{\rho} = 0$. Thus $\rho_0 f_n = y_n = \pm 1$ and the margins for all n are the same. We assume that this is the minimum ρ at convergence with $\lambda = 0$ and thus with no restrictions on rank. The associated ρ is ρ_0. This also implies that there is no other solution for $\lambda = 0$ with f_n that are *all equally larger* than $\frac{1}{\rho}$.
Let us now assume that $\lambda > 0$. Consider two extreme situations.

- The size of the minibatch B is $B = N$. There is effectively no rank constraint wrt the N data points of a minibatch. Assume the margins all equal to each other as $\bar{f}_n = \mu = \frac{1}{\rho_0}, \quad \forall n, \sigma = 0$ and thus $M = \mu^2 = \frac{1}{\rho_0^2}$. At equilibrium of SGD then $\dot{\rho} = 0 = 2N\mu - 2N\rho M - 2\lambda\rho$ which yields

$$\rho_\lambda = \frac{\rho_0}{1 + \rho_0^2 \frac{\lambda}{N}}. \tag{46}$$

- The size of the minibatch B is $B << N$. In this case

$$\rho_\lambda = \frac{\rho_0}{1 + \rho_0^2 \frac{\lambda}{B}}, \tag{47}$$

  but this only holds for the B points of the minibatch.

E Unnormalized vs Normalized dynamics

As shown in E.1, the equilibria with and without normalization are the same for ρ and V_k but the dynamics is different. Consider Figure 1. Assume that the network on Figure 1a is un-normalized, that is optimized via GD without Lagrange multipliers and the network in Figure 1b is normalized, that is optimized via GD with the Lagrange multiplier term. For 1a, consider, for simplicity, the case in which all the norms ρ_k of the weight matrices $1, \cdots, L-1$ are initialized with the same value. Then because

[13] Numerical simulations show that even for linear degenerate networks convergence is independent of initial conditions only if $\lambda > 0$. In particular, normalization is then effective at ρ_0, unlike the $\lambda = 0$ case.

of Lemma 10 all the $\rho_k, \quad \forall k = 1, \cdots, L-1$ of Figure 1a will change together and remain equal to each other. Let us call $\rho_k = \rho_1$. It is then possible to consider ρ for the network of Figure 1b as $\rho = \rho_1^L$ (because of the homogeneity of the ReLUs) and look at its dynamics. Consider the case of $\lambda = 0$.
The equations for the un-normalized case are then

$$\dot{\rho} = 2L\rho^{\frac{2L-2}{L}} \left[\sum_n \bar{f}_n - \sum_n \rho(\bar{f}_n)^2 \right] \tag{48}$$

and

$$\dot{V}_k = -2\rho^{\frac{L-2}{L}} \sum_n \left(1 - \rho \bar{f}_n\right) \cdot \left(\frac{\partial \bar{f}_n}{\partial V_k} - V_k \bar{f}_n \right). \tag{49}$$

The equations for the normalized case (Figure 1b), in which we have a single ρ parameter, are

$$\dot{\rho} = 2[\sum_n \bar{f}_n - \sum_n \rho(\bar{f}_n)^2] \tag{50}$$

and

$$\dot{V}_k = 2\rho \sum_n \left[(1 - \rho \bar{f}_n)(V_k \bar{f}_n - \frac{\partial \bar{f}_n}{\partial V_k}) \right]. \tag{51}$$

Recall that for $\lambda = 0$, ρ_0 corresponds to the inverse of the margin: thus $\frac{1}{\rho_0} = f_n$, since f_n is the same for all n. Thus $\dot{V}_k$ is proportional to ρ (ρ is the inverse of the margin) in the normalized case and to $\rho^{\frac{L-2}{L}}$ in the un-normalized case. The proportionality factor 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 good strategy, since large margin corresponds to "good" minima in terms of generalization (for classification).

E.1 Unnormalized dynamics

We consider the dynamical system induced by GD on a deep net with ReLUs (see Figure 1a). We change variables by using [14] $W_k = \rho_k V_k$, $\|V_k\| = 1$. Following the calculations in [4], the following identities hold: $\frac{\partial \rho_k}{\partial W_k} = V_k^T$ and $\frac{\partial g_n}{\partial W_k} = \frac{\rho}{\rho_k} \frac{\partial f_n}{\partial V_k}$.
Thus gradient descent on $L = \mathcal{L}_S = \sum_n (\rho f_n - y_n)^2$ with the definitions of V_k and ρ_k yields the dynamical system (with $\dot{W}_k = -\frac{\partial L}{\partial W_k}$)

$$\dot{\rho_k} = \frac{\partial \rho_k}{\partial W_k} \dot{W}_k = V_k^T \dot{W}_k = -2 \sum_n (\rho_k^L f_n - y_n) f_n \rho_k^{L-1} = -2\rho_k^{L-1} [\sum_n \rho_k^L (f_n)^2 - \sum_n f_n y_n] \tag{52}$$

and, with $S_k = I - V_k V_k^T$,

$$\dot{V}_k = \frac{\partial V_k}{\partial W_k} \dot{W}_k = \frac{S_k}{\rho_k} \dot{W}_k = -2 \frac{\rho}{\rho_k^2} \sum_n (\rho f_n - y_n)(\frac{\partial f_n}{\partial V_k} - V_k f_n). \tag{53}$$

E.2 Equal growth

If we assume that all the ρ_k are the same at initialization, we can use the following lemma to show that all ρ_k are the same at all times :

Lemma 10 [4] *$\frac{\partial \rho_k^2}{\partial t}$ is independent of k for $\lambda = 0$ and no normalization.*

Proof
Start from $\mathcal{L}_S = \frac{1}{N} \sum_n (1 - \rho \bar{f}_n)^2 + \lambda \sum_k \|W_k\|^2$ with $\lambda = 0$. Then

$$\dot{W}_k = -\frac{2}{N} \sum_n (1 - \rho \bar{f}_n) \rho \frac{\partial \bar{f}_n}{\partial W_k} \tag{54}$$

[14] Changing coordinates from W_k to V_k, we can convert the previous dynamical system to one in ρ, V_k. Using some basic vector calculus to get $\frac{\partial \rho_k}{\partial t} = \frac{1}{\rho_k} \left\langle W_k, \frac{\partial W_k}{\partial t} \right\rangle$ and $\frac{\partial V_k}{\partial t} = \frac{1}{\rho_k} \left(I - V_k V_k^\top \right) \frac{\partial W_k}{\partial t}$.

and, since $\|\dot{W_k}\|^2 = W_k\dot{W_k}$, we write

$$\|\dot{W_k}\|^2 = -\frac{2}{N}\sum_n(1-\rho\bar{f}_n)\rho\bar{f}_n \tag{55}$$

which is independent of k.
Notice that if $\lambda > 0$, then for weight decay applied to all layers (that is with a regularization term of the form $\lambda\sum_k \rho_k^2$)

$$\|\dot{W_k}\|^2 = -\frac{2}{N}\sum_n(1-\rho\bar{f}_n)\rho\bar{f}_n - \lambda W_k \tag{56}$$

Thus the equal growth lemma 10 does not hold in the case of weight decay. Furthermore it does not hold in the case that the regularization term is $\lambda\rho^2$ with $\rho = \Pi_k\rho_k$.

E.3 Equal weight norms at all layers

Assume the setup of the previous section. Then $\rho = \rho_k^L$, where L is the number of layers.
We use Equation (52) to derive the dynamics of $\rho = \rho_k^L$ in terms of $\dot{\rho} = \sum_k \frac{\partial\rho}{\partial\rho_k}\dot{\rho_k}$. Thus,

$$\dot{\rho} = 2L\rho^{\frac{2L-2}{L}}[\sum_n f_n y_n - \sum_n \rho(f_n)^2] \tag{57}$$

which is an equation of the type known as "differential logistic equation" used for instance to model sigmoidal population growth. It has an interesting dynamics as shown in the simplified simulations of Figure 12. The equilibrium value for $\dot{\rho_k} = 0$ is

$$\rho_0 = \frac{\sum_n \bar{f}_n}{\sum_n f_n^2}. \tag{58}$$

Similarly, for V_k:

$$\dot{V_k} = -2\rho^{\frac{L-2}{L}}\sum_n(\rho f_n - y_n)(\frac{\partial f_n}{\partial V_k} - V_k f_n). \tag{59}$$

At equilibrium for V_k – that is when $\dot{V_k} = 0$ – the equation gives (with $\ell_n = \rho f_n - y_n$ and assuming $\sum f_n\ell_n \neq 0$)

$$\sum_n(\rho f_n - y_n)\frac{\partial f_n}{\partial V_k} = \sum_n(\rho f_n - y_n)(V_k^0 f_n). \tag{60}$$

F Extending the Analysis to GD

Recently, [54] showed that a natural approximation to gradient descent within a continuous gradient flow formulation is equivalent to adding to the loss functional $\mathcal{L}_S$ a term proportional to $\frac{\eta}{4}$, consisting of the norm square of the gradient of $\mathcal{L}_S$. This is equivalent (see [55]) to replacing in the gradient flow equation terms like $\dot{x}$ with terms that are $\frac{\eta}{2}\ddot{x} + \dot{x}$. The informal explanation is that the gradient descent term $x(t+\eta) - x(t) = -\eta F$ can be approximated by expanding $x(t+\eta)$ in a Taylor series for small η to a quadratic approximation, that is $x(t+\eta) \approx x(t) + \eta\dot{x}(t) + \frac{\eta^2}{2}\ddot{x}(t)$. Thus the gradient descent equation becomes $\dot{x}(t) + \frac{\eta}{2}\ddot{x}(t) = -F$.
With this approximation Equations (2) become (taking into account that $\mathcal{L}_S = \sum_n(1-\rho\bar{f}_n)^2 + \nu\sum_{k=1}^{L-1}\|V_k\|^2 + \lambda\rho^2$)

$$\frac{\eta}{2}\ddot{\rho} + \dot{\rho} = 2[\sum_n(1-\rho\bar{f}_n)\bar{f}_n] - 2\lambda\rho \tag{61}$$

$$\frac{\eta}{2}\ddot{V_k} + \dot{V_k} = 2\rho\sum_n[(1-\rho\bar{f}_n)(V_k\bar{f}_n - \frac{\partial\bar{f}_n}{\partial V_k})] \quad \forall k \leq L. \tag{62}$$

We observe immediately that the equation in ρ – since it has the form $\eta\ddot{\rho} + \dot{\rho} + 4\lambda\rho - 4C = 0$ may show oscillations(the frequency of the "undamped oscillations" is $\sqrt{\frac{4(\frac{1}{N}\sum \bar{f}_n^2 + 2\lambda)}{\eta}}$. Whenever $1 > \sqrt{\frac{2}{N}\sum_n \bar{f}_n^2 + 2\lambda}$, the linearized dynamics is the dynamics of a damped oscillator. However, we didn't find empirical evidence for such oscillations examining the power spectrum.
By using $W_L = \rho V_L$ (see figure 1), we replace this system of equations with the following system

$$\forall\, k < L: \ \frac{\eta}{2}\ddot{V}_k + \dot{V}_k = 2\rho \sum_n [(1 - \rho\bar{f}_n)(V_k \bar{f}_n - \frac{\partial \bar{f}_n}{\partial V_k})]. \tag{63}$$

$$\frac{\eta}{2}\ddot{W}_L + \dot{W}_L = 2\sum_n [(1 - \overline{g}_n)\frac{\partial \overline{g}_n}{\partial W_L})] - 2\lambda W_L. \tag{64}$$

As a sanity check we see that $W^T\dot{W} = \rho\dot{\rho}$ since $W_L = \rho_L V_L$, $\|V_L\| = 1$, $g_n = \rho f_n$.
We multiply the last equation on the left by W^T obtaining

$$\frac{\eta}{2}W^T\ddot{W}_L + W^T\dot{W}_L = 2\sum_n (1 - \overline{g}_n)\overline{g}_n - 2\lambda W_L^2 \tag{65}$$

We change variables in the last equation:

$$\frac{\eta}{2}W^T\ddot{W}_L + W^T\dot{W}_L = 2\rho\sum_n [(1 - \rho\bar{f}_n)\bar{f}_n) - 2\lambda\rho^2 \tag{66}$$

Since $\dot{W}_L = \rho\dot{V}_L + \dot{\rho}V_L$ and $\ddot{W}_L = 2\dot{\rho}\dot{V}_L + \rho\ddot{V}_L + \ddot{\rho}V_L$, the last equation becomes for $\dot{\rho} = 0$

$$\rho V^T\frac{\eta}{2}\rho\ddot{V}_L + \rho V^T\rho\dot{V}_L = 2\rho\sum_n [(1 - \rho\bar{f}_n)\bar{f}_n) - 2\lambda\rho^2 \tag{67}$$

Now notice the following simple relation between accelerations and velocities: $\frac{\partial(V^TV)}{\partial t} = 2V^T\dot{V}$ and $\frac{\partial^2(V^TV)}{\partial t^2} = 2(\dot{V}^T\dot{V} + V^T\ddot{V}) = 2(\|\dot{V}\|^2 + V^T\ddot{V})$. If the norm $\|V_L\|$ is constant, then $\frac{\partial^2(V^TV)}{\partial t^2} = 0$. It follows $V^T\ddot{V} = -\|\dot{V}\|^2$.
Thus

$$\rho^2 V^T\frac{\eta}{2}\ddot{V}_L = 2\rho\sum_n ((1 - \rho\bar{f}_n)\bar{f}_n - 2\lambda\rho^2 \tag{68}$$

$$-\frac{\eta}{2}\rho\|\dot{V}_L\|^2 = 2\sum_n (1 - \rho\bar{f}_n)\bar{f}_n - 2\lambda\rho \tag{69}$$

Equation (61) implies that when $\dot{\rho} = 0$ then $\|\dot{V}_L\|^2 = 0$.

G BN, GD, LN: remarks

G.1 Remarks on BN

Without BN and without WD $\frac{\partial g_n(W)}{\partial W_k} = \frac{\rho}{\rho_k}\frac{\partial f_n(V)}{\partial V_k}$; with BN but without weight decay this becomes $\frac{\partial g_n(W)}{\partial W_k} = \rho\frac{\partial f_n(V)}{\partial V_k}$, $\quad \forall k < L$ and $\frac{\partial g_n(W)}{\partial W_L} = \frac{\partial f_n(V)}{\partial V_k}$.
This dynamics can also be written as $\dot{\rho}_k = V_k^T\dot{W}_k$ and $\dot{V}_k = \rho S\dot{W}_k$ with $S = I - V_kV_k^T$. This shows that if $W_k = \rho_k V_k$ then $\dot{V}_k = \frac{1}{\rho_k}\dot{W}_k$ as mentioned in [35].

G.2 Lagrange multiplier vs Batch Normalization

The constrained Lagrange dynamics will not change the initial norm of the $L-1$ layers: to ensure that $\|V_k\| = 1$ the initial value of the $L-1$ weight matrices must be $\|V_k\| = 1, \quad k = 1, \cdots, L-1$.
In our toy model the dynamics above with Lagrange multipliers captures what is commonly believed to be the key normalization property of batch normalization. This property can be summarized by the fact that the output of unit i given by $(Wh)_i$, where h is the vector of activities of the units in the previous layer, does not change after normalization if W is replaced by scaled version cW. It is

important to emphasize, however, that the Lagrange dynamics does not reflect several aspects of Batch Normalization. In particular, in our model the weight matrices at each layer are normalized whereas in Batch Normalization each unit activity in each layer is normalized across the batch. Thus the dynamics of the weights in our model is different from the dynamics under Batch Normalization (see discussions in [4] and also [35]).
Notice also that in the model of Figure 1b, we regularize a single ρ at the top of the network, whereas in the standard usage of BN each layer norm $\rho_k, \quad \forall k = 1, \cdots, L$ is subject to weight decay. All layers $k = 1, \cdots, L-1$, but the last one, are also subject to BN.
In summary, the Lagrange multiplier normalization is equivalent[4] to Weight Normalization but is different in several aspects wrt Batch Normalization. We believe however that both capture a key property of normalization techniques – the invariance with respect to scaling the weight matrices.

G.3 Normalization at each layer

In our model, we considered the situation of Figure 1b, where only the top layer weight matrix is not normalized and has ρ at the top, subject to weight decay, while the previous layers are all normalized but are not subject to weight decay. This model neatly separates layers that are normalized from layers that have weight decay; in this model no layer has weight decay *and* normalization.
However, in normal practice of training a deep network, the weight matrices W_k in all layers up to layer $L-1$ are subject to weight decay and normalization via batch norm; only the last layer weights W_L are not normalized but still subject to weight decay. In this section, we consider this case, using Lagrange multipliers in place of batch norm.
The usual training of a deep net as in Figure 1a corresponds to minimizing the functional

$$\mathcal{L}_{\mathcal{S}} = \sum_n (1 - \rho \bar{f}_n)^2 + \sum_{k=1}^{L-1} \nu_k \rho_k^2 (\|V_k\|^2 - 1) + \mu(\|V_L\|^2 - 1) + \lambda \sum_{k=1}^{L} \rho_k^2 \tag{70}$$

with $\rho_k^2 \|V_k\|^2 = 1$ for $k = 1, \cdots, L$ with the definition $\rho = \Pi_k \rho_k$. Notice that Equation (70) is different from Equation (1) for the toy model.

G.3.1 Gradient flow

Gradient flow in the model of Figure 1 is

$$\dot{\rho_k} = -\frac{\partial \mathcal{L}}{\partial \rho_k} = 2\frac{\rho}{\rho_k} \sum_n (1 - \rho \bar{f}_n) \bar{f}_n - 2\nu_k \rho_k \|V_k\|^2 - 2\lambda \rho_k \quad \forall k = 1, \cdots, L-1 \tag{71}$$

$$\dot{\rho_L} = -\frac{\partial \mathcal{L}_{\mathcal{S}}}{\partial \rho_L} = 2\frac{\rho}{\rho_L} \sum_n (1 - \rho \bar{f}_n) \bar{f}_n - 2\lambda \rho_L \tag{72}$$

and for $k < L$

$$\dot{V_k} = -\frac{\partial L}{\partial V_k} = 2\rho \sum_n (1 - \rho \bar{f}_n) \frac{\partial \bar{f}_n}{\partial V_k} - 2\nu \rho_k^2 V_k \tag{73}$$

and

$$\dot{V_L} = -\frac{\partial L}{\partial V_L} = 2\rho \sum_n (1 - \rho \bar{f}_n) \frac{\partial \bar{f}_n}{\partial V_L} - 2\mu V_L \tag{74}$$

To find μ we multiply by V_L^T to get $0 = 2\rho \sum_n (1 - \rho \bar{f}_n) \bar{f}_n - 2\mu \|V_L\|^2$ which gives

$$\mu = \frac{\rho}{\|V_L\|^2} \sum_n (1 - \rho \bar{f}_n) \bar{f}_n. \tag{75}$$

To find ν_k we multiply by V_k^T and obtain $0 = \rho \sum_n (1 - \rho \bar{f}_n) \bar{f}_n - \nu_k \|V_k\|^2 \rho_k^2$ which gives

$$\nu_k = \frac{\rho}{\|V_k\|^2 \rho_k^2} \sum_n (1 - \rho \bar{f}_n) \bar{f}_n. \tag{76}$$

Thus the gradient flow is the following dynamical system for $k < L$

$$\dot{\rho_k} = -2\lambda \rho_k \quad \forall k = 1, \cdots, L-1 \tag{77}$$

$$\dot{V}_k = 2\rho \sum_n (1-\rho \bar{f}_n)(\frac{\partial \bar{f}_n}{\partial V_k} - V_k \bar{f}_n) \quad \forall k = 1, \cdots, L-1 \tag{78}$$

$$\dot{\rho_L} = 2\frac{\rho}{\rho_L} \sum_n (1-\rho \bar{f}_n)\bar{f}_n - 2\lambda \rho_L \tag{79}$$

$$\dot{V}_L = 2\rho \sum_n (1-\rho \bar{f}_n)(\frac{\partial \bar{f}_n}{\partial V_L} - V_L f_n) \tag{80}$$

The solution of Equation (77) is $\rho(t) = 1 - (1-\rho_{t=0})e^{-2\lambda t}$ because $\rho^2 = 1$ for $t \to \infty$.
An equivalent alternative and simpler formulation is to start from

$$\mathcal{L}_S = \sum_n (1-\overline{g}_n)^2 + \sum_{k=1}^{L-1} \nu_k \|W_k\|^2 + \lambda \sum_{k=1}^{L} \|W_k\|^2 \tag{81}$$

under the constraint $\|W_k\|^2 = 1$.
Gradient flow in this model is

$$\dot{W}_k = 2\sum_n (1-\overline{g}_n)\frac{\partial \overline{g}_n}{\partial W_k} - 2\nu_k W_k - 2\lambda W_k \quad \forall k = 1, \cdots, L-1 \tag{82}$$

and

$$\dot{W}_L = 2\sum_n (1-\overline{g}_n)\frac{\partial \overline{g}_n}{\partial W_L} - 2\lambda W_L \tag{83}$$

To find ν_k we multiply by W_k^T and obtain $0 = \sum_n (1-\rho\overline{g}_n)\overline{g}_n - \nu_k - \lambda$ which gives

$$\nu_k = \sum_n (1-\rho\overline{g}_n)\overline{g}_n - \lambda. \tag{84}$$

Thus the gradient flow in W_k is the following dynamical system for $k < L$

$$\dot{W}_k = 2\sum_n (1-\overline{g}_n)(\frac{\partial \overline{g}_n}{\partial W_k} - W_k\overline{g}_n) \tag{85}$$

and

$$\dot{W}_L = 2\sum_n (1-\overline{g}_n)\frac{\partial \overline{g}_n}{\partial W_L} - 2\lambda W_L \tag{86}$$

This dynamics corresponds to the dynamics in ρ and V_k of section 4.

G.3.2 Gradient Descent

Here we take into account the effect of discretization for Figure 1a as we did in a previous section with η terms added to the left-hand side of the previous set of equations obtaining

$$\frac{\eta}{2}\ddot{W}_k + \dot{W}_k = 2\sum_n (1-\overline{g}_n)(\frac{\partial \overline{g}_n}{\partial W_k} - W_k\overline{g}_n) \tag{87}$$

and

$$\frac{\eta}{2}\ddot{W}_L + \dot{W}_L = 2\sum_n (1-\overline{g}_n)\frac{\partial \overline{g}_n}{\partial W_L} - 2\lambda W_L \tag{88}$$

H Experiments on dynamics and generalization

We first look at a typical sequence of events during training with LN. The legend of Figure 13 summarizes it.

Next, we consider again the effect of label noise on ρ and on expected error. As discussed in the main text, 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. This also happens with Batch Normalization as shown in Figure 14. These results are fully consistent with the generalization bounds Equations 39 and 40.

However, the margin does not explain the behavior shown in Figure 15 where small differences in margin are actually anti-correlated with small differences in test error. Tighter bounds (see [50]) may explain these small effects. Alternatively, if there exist several almost-interpolating solutions with the same norm ρ_0, they may have similar norm and similar margin but different ranks of the weight matrices.

I Remarks on rank constraints in SGD

- *Convolutional layers.* Notice that the operation computed by a convolutional layer is equivalent to a linear transformation with a Toeplitz matrix. Since non-zero Toeplitz matrices cannot have rank one, the low rank argument, as we described it above, does not immediately apply to convolutional layers. However, a similar result applies to convolutional layers, when considering the matrix obtained by turning the filters into a matrix (Galanti and Siegel, in preparation).

- *Intuition.* A common intuition for why there should be a bias towards low rank of any layer added to an interpolating network is that the network is biased to reach the maximum $y_n f_n$ for each training example n with the minimum ρ because of regularization. This leads to *an implicit minimization of the "stable rank"*, defined as the ratio of the square of the Frobenius norm of each additional weight matrix and the square of its spectral norm $\frac{||V||_F^2}{||V||^2} = \frac{\sum_j \sigma_j^2}{\max_i \sigma_i^2}$, as described in [10]. This argument, however, is a separate mechanism wrt the low rank bias of SGD described earlier. The relation between the two mechanisms is so far unclear.

- *LM normalization.* The previous results were derived assuming normalization by Lagrange multipliers. It turns out [43] that for the square loss normalization is not needed (as it is easy to verify) to yield SGD noise. However, for an exponential type loss, LM normalization can replace regularization wrt to SGD noise and rank constraint. The theoretical prediction is confirmed by experiments (compare Figure 5 and Figure 16).

J Low Rank Constraint for Exponential-Type Loss Functions

Assume that a deep multilayer neural network $f(x) = W^L \sigma(W^{L-1} \dots \sigma(W^1 x))$ is trained on a binary classification task with weight normalization with respect to an exponential loss function without weight decay. The stationary points of the SGD flow of the normalized weight matrices V_k are given (see [7, 2]) for any finite $\rho = \prod_{k=1}^L \rho_k$, where $\rho_k = \|W_k\|_2$ by

$$\sum_n^B e^{-\rho y_n \hat{f}_n} y_n \left(\frac{\partial \hat{f}_n}{\partial V_k} - V_k \hat{f}_n \right) = 0, \tag{89}$$

where $\hat{f}_n$ is the scalar output of the ReLU network with normalized weight matrices when the input is x_n, y_n is the corresponding binary label and B is the size of the minibatches. For any large but finite value of ρ, the equation shows a similar rank constraint (in fact rank zero, see [43]) on the V_k weight matrices, *even with* $\lambda = 0$. Figure 16 shows that indeed, unlike the square loss case, there is noise in the margin, independently of the presence or absence of regularization, as predicted by our analysis.

List of Figures

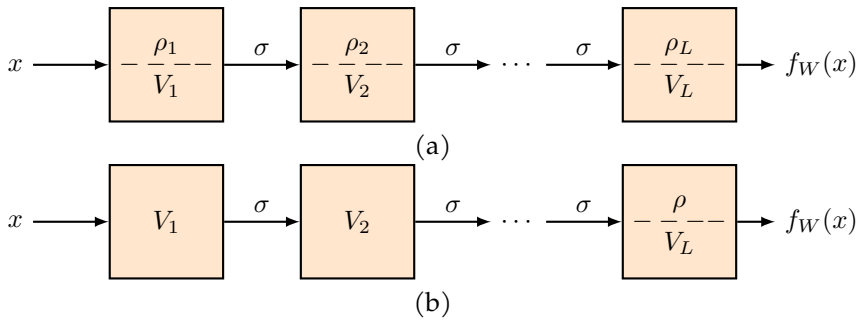


Figure 1: *An illustration of two parametrizations of $f_W(x)$. In (a) we decompose each layer's weight matrix W_i into its norm ρ_i and its normalized version V_i. In (b) we normalize each layer except for the top layer's matrix W_L that is decomposed into a global ρ and the last layer V_L. Normalizing the weight matrices, as weight normalization (equivalent to LN) does, is different from Batch Normalization, though both normalization techniques capture the relevant property of normalization – to make the dot product invariant to scale.*

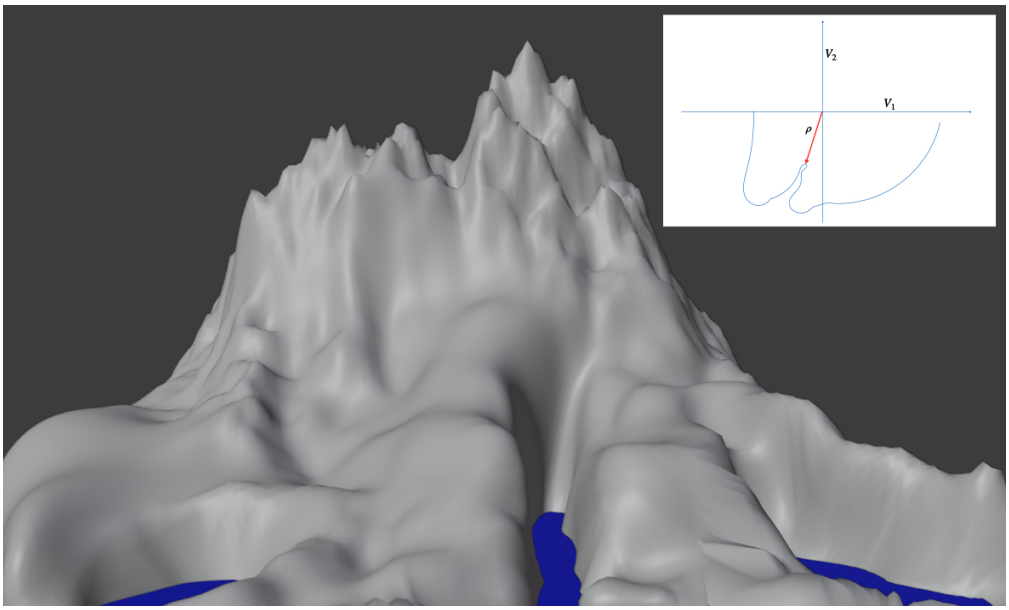
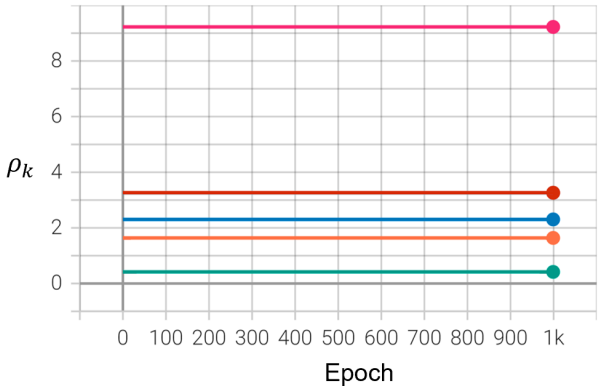


Figure 2: *A speculative view of the landscape of the loss with global degenerate valleys for $\rho \geq \rho_0$ with V_1 and V_2 weights of unit norm. Think of the loss as the mountain emerging from the water with zero-loss being the level of the water. ρ is the radial distance from the center of the mountain as shown in the inset. The coastline of the loss marks the boundary of the zero loss degenerate minimum where $\mathcal{L} = 0$ in the high-dimensional space of ρ and $V_k \quad \forall k = 1, \cdots, L$. The degenerate global minimum is shown here as a connected valley outside the coastline. The red arrow marks the minimum loss with minimum ρ. Notice that, depending on the shape of the multidimensional valley, regularization may not guarantee convergence to the minimum norm solution, unlike in the linear network case*



Color	Layer Name	ρ_k	Epoch No.
	Conv 1	1.639	999
	Conv 2	2.304	999
	Conv 3	3.262	999
	Conv 4	3.261	999
	FC 1	9.225	999
	FC 2	0.4179	999

Figure 3: *Training dynamics of ρ_k during model (b) training with the Lagrange Multiplier normalization over 1000 epochs. The model contains four convolutional layers, two fully connected layers and the top ρ (a learnable scalar parameter that can be initialized with different values). $\rho_k(k \in [L-1])$ are effectively stable during training because of weight normalization. The number of channels for the four convolutional layers (Conv1∼Conv4) are 32, 64, 128 and 128, the filter size is 3×3, the hidden sizes of the last two fully connected layers (FC1 and FC2) are 1024 and 2, respectively. As mentioned in the text the norms of the convolutional layers is just the norm of the filters.*

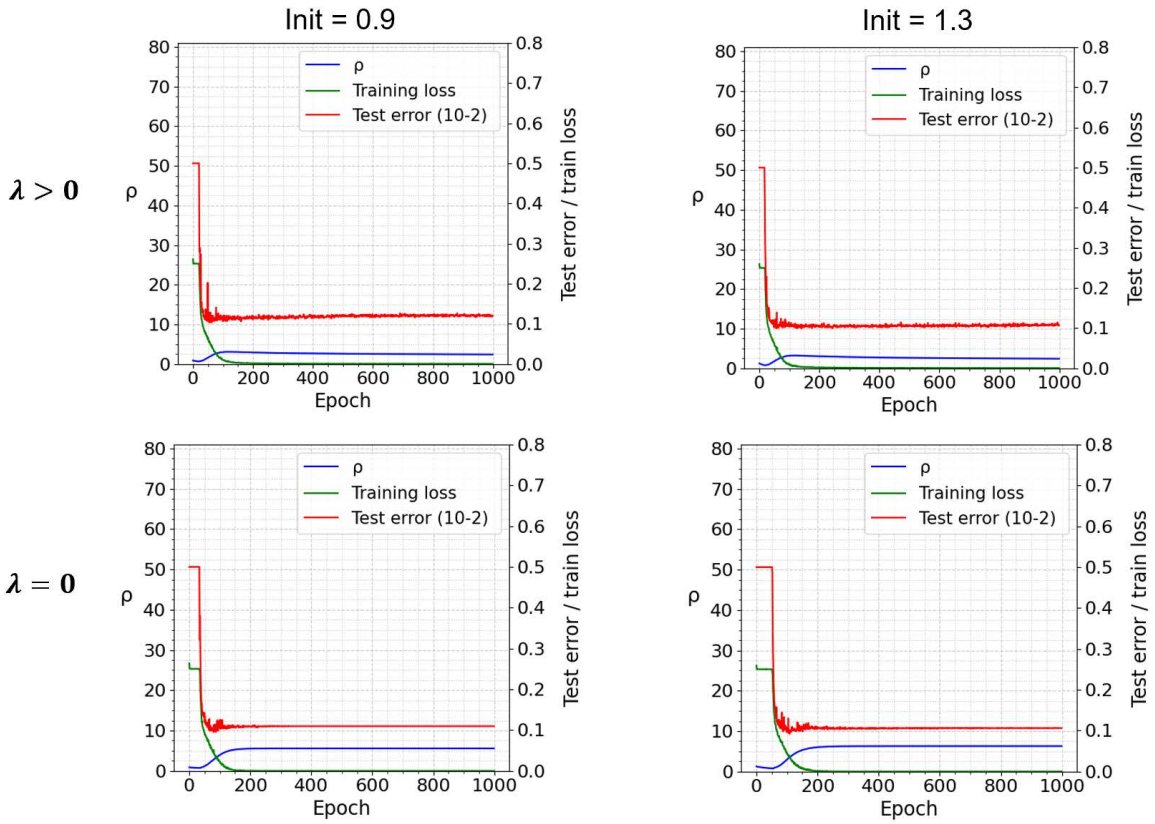


Figure 4: *Training dynamics of last layer norm ρ, training loss and test error over 1000 epochs with different initialization (0.9) in the first column and (1.3) in the second column. The first row is with Weight Decay $\lambda = 0.001$, and the second row is with Weight Decay $\lambda = 0$. The network was trained with Cosine Annealing learning rate scheduler $(\text{with initial learning rate } \eta = 0.03, \text{ending with } \eta = 0.0299)$.*

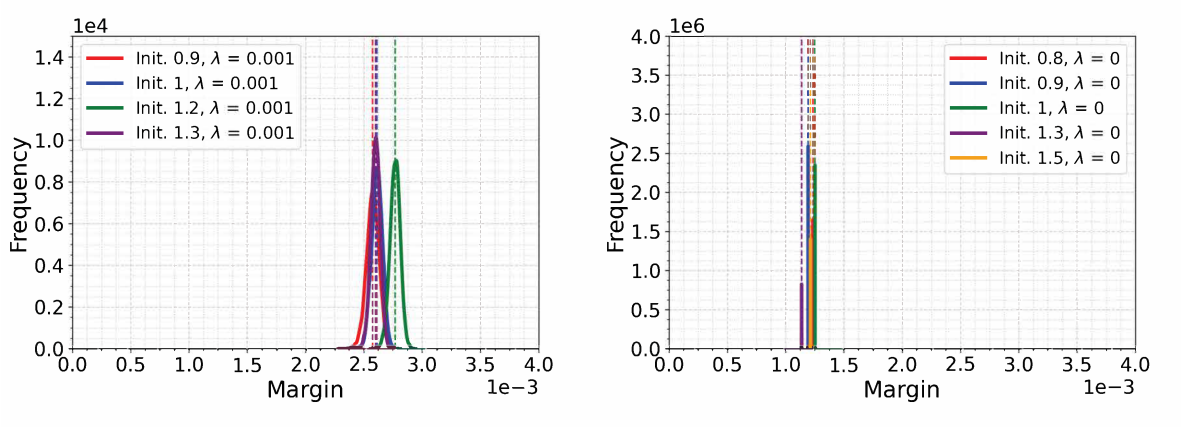


Figure 5: *Training margins computed over 10 runs for binary classification on CIFAR10 trained with square loss, Lagrange Multiplier normalization, and Weight Decay* $(\lambda) = 0.001$ *(left) and without Weight Decay (right,* $\lambda = 0$*) for different initializations* $(init. = 0.8, 0.9, 1, 1.2, 1.3$ *and* $1.5)$ *with SGD and minibatch size of* 128*. The margin distribution is Gaussian-like with standard deviation* $\approx 10^{-4}$ *over the training set* $(N = 10^4)$*. The margins without Weight Decay result in a range of smaller margin values, each with essentially zero variance. As mentioned in the text the norms of the convolutional layers is just the norm of the filters.*

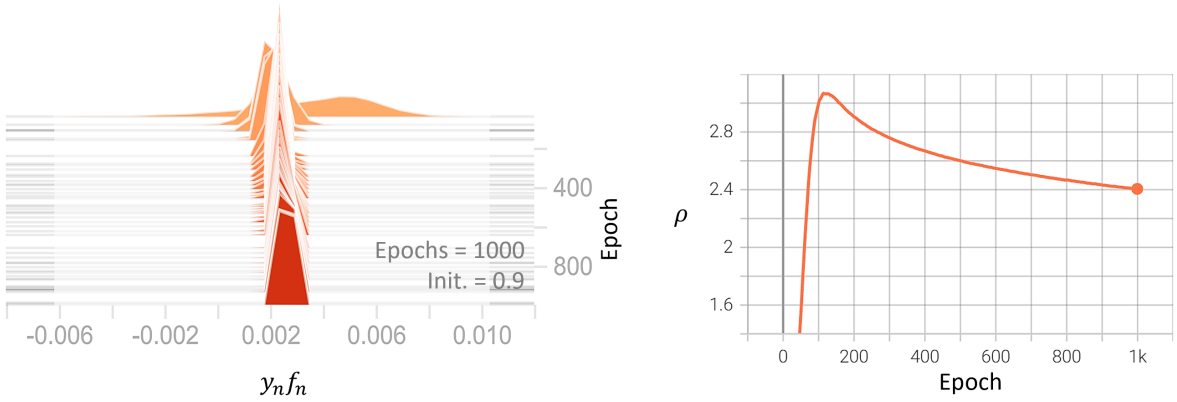


Figure 6: *Histogram of $y_n f_n$ across 1000 training epochs for binary classification on the CIFAR10 dataset with Lagrange Multiplier and weight decay $(\lambda) = 0.001$, initial learning rate 0.03, initialization 0.9. The histogram narrows as training progresses. The final histogram (in red) is concentrated, as expected for the emergence of NC1. The right side of the plot shows the time course of the top ρ over the same 1000 epochs.*

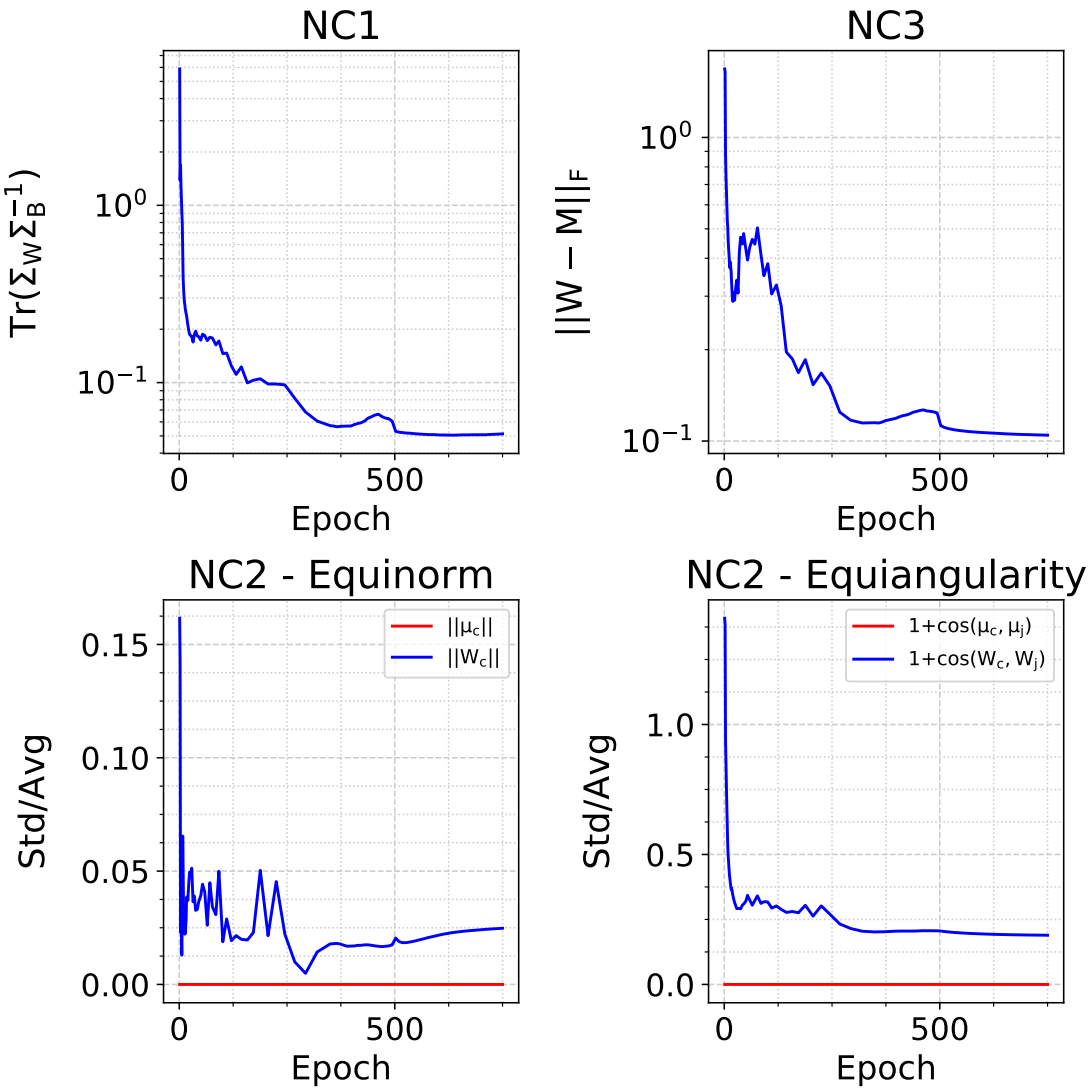


Figure 7: *Neural Collapse occurs during training for binary classification. The key conditions for Neural Collapse are: (i) NC1 - Variability collapse, which is measured by* $Tr(\Sigma_W \Sigma_B^{-1})$*, where* Σ_W, Σ_B *are the within and between class covariances, (ii) NC2 - equinorm and equiangularity 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, and (iii) NC3 - Self-duality or the distance between the normalized classifiers and mean features. This network was trained on two classes of CIFAR10 with Weight Normalization and Weight Decay = 5e-4, learning rate 0.067, for 750 epochs with a stepped learning rate decay schedule.*

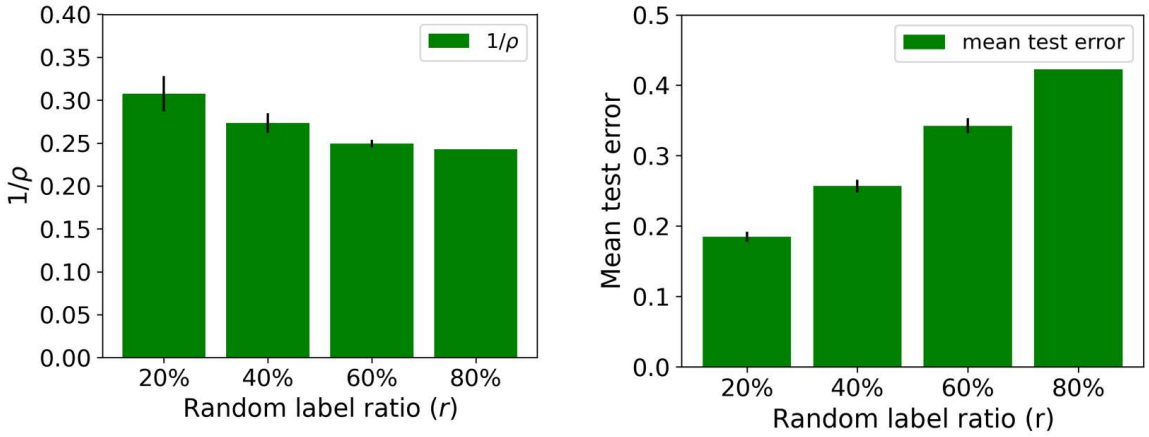


Figure 8: *Mean $1/\rho$ and test error results over 10 runs for binary classification on CIFAR10 trained with Lagrange Multiplier and different percentages of random labels ($r = 20\%, 40\%, 60\%$ and 80%), initialization scale 1 and weight decay 0.001. As mentioned in the text the norm of the convolutional layers is just the norm of the filters.*(*Note that this network fails to get convergence with 100% random labels.*)

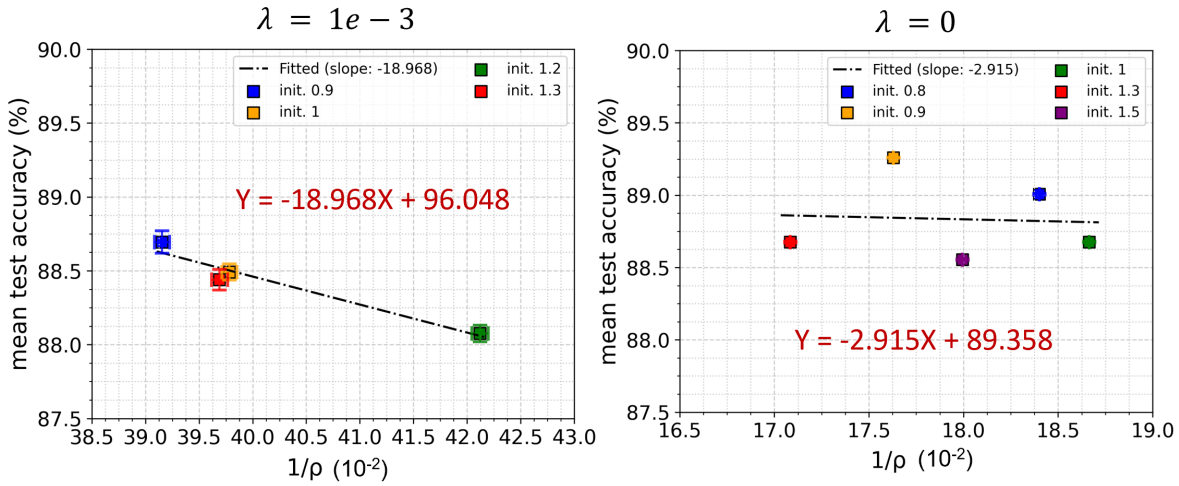
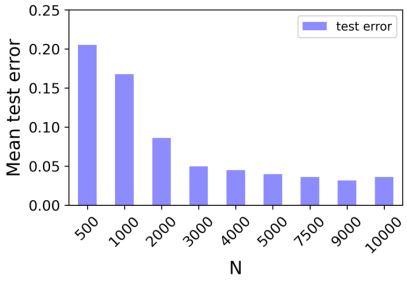


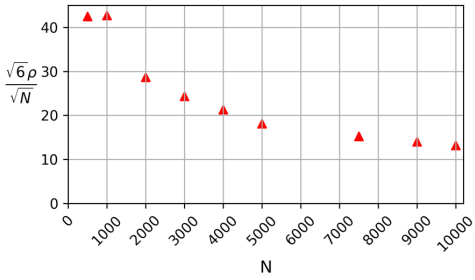
Figure 9: *Scatter plots for $1/\rho$ and mean test accuracy based on 10 runs for binary classification on CIFAR10 using Lagrange Multiplier normalization (LN), square loss and Weight Decay (left) and without Weight Decay (right). In the left figure, the network was trained with different initialization scales (init. = [0.9, 1, 1.2, 1.3]) and with weight decay ($\lambda = 1e-3$), while in the right figure, the network was trained with init. = [0.8, 0.9, 1, 1.3, 1.5] and no weight decay ($\lambda = 0$). The horizontal and vertical error bars correspond to the standard deviations of $1/\rho$ and mean test accuracy computed over 10 runs for different initializations, while the square dots correspond to the mean values. When $\lambda > 0$, the coefficient (R^2), p-value and slope for linear regression between $1/\rho$ and mean test accuracy are: $R^2 = 0.94$, p-value = 0.031, slope = -18.968; When $\lambda = 0$, the coefficient $R^2 = 0.004$, p-value = 0.92 and the slope = -2.915.*

N	total ρ	test error	$\frac{\sqrt{6}\rho}{\sqrt{N}}$
500	387.672	0.205	42.467
1000	551.282	0.168	42.702
2000	523.698	0.086	28.684
3000	544.018	0.050	24.329
4000	550.868	0.045	21.335
5000	522.870	0.040	18.113
7500	540.055	0.036	15.275
9000	544.738	0.032	14.065
10000	538.174	0.036	13.183

(a)



(b)



(c)

Figure 10: Product norm (ρ) and test error with respect to different training data sizes (N) for the six-layer model trained with LM and square loss. The initialization scale is 0.1, weight decay $\lambda = 10^{-3}$, no biases, the initial learning rate is 0.03 with cosine annealing scheduler; we used a SGD optimizer (momentum $= 0.9$), test data size $= 2000$ in a binary classification task on CIFAR10 dataset. (a) The table shows the product norm ρ, mean test errors (average over the last 100 epochs), and generalization upper bound for different N (see Equation **??**). (b) A bar plot for the mean test errors by different N. (c) Generalization error upper bound defined as ($\frac{\sqrt{L}\rho}{\sqrt{N}}$, where $L = 6$) for different N. The bounds are vacuous but "only" by an order of magnitude, while other bounds based on the number of parameters (here 3519335) are typically much looser.

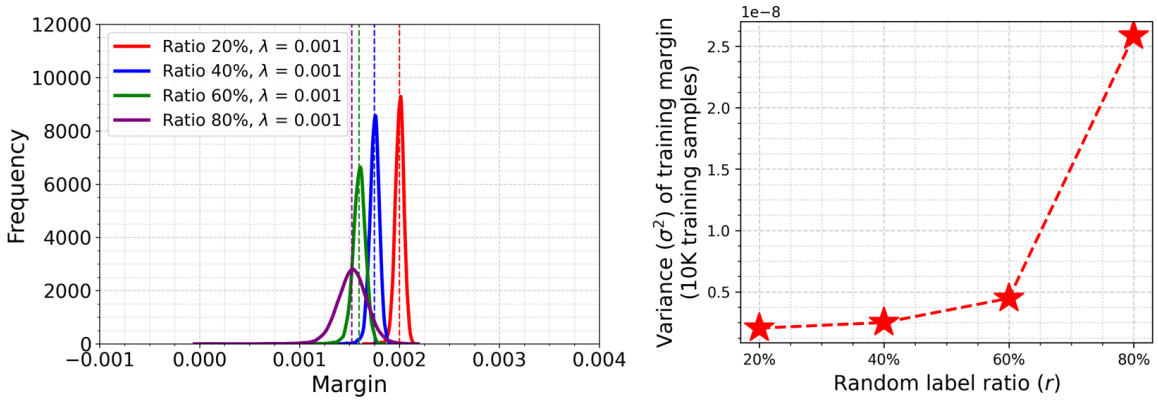


Figure 11: *Average training margin distribution (left) and the corresponding margin variance - σ^2 (right) over 10K training samples by 10 runs for binary classification on the CIFAR10 with four different random label ratios (r = 20%, 40%, 60% and 80%). The networks were trained with Lagrange Multiplier, weight decay ($\lambda = 1e-3$) and initialization scale 1. The higher the random label ratio is, the smaller the mean margins (vertical dashed lines in the left figure) and the greater the variance (right) of the training margins. Specifically, the mean margin (μ) decreases from 2.001e-3 (with 20% random labels) to 1.524e-3 (with 80% random labels); the variance (σ^2) grows from 2.04e-9 (with 20% random labels) to 2.58 e-8 (with 80% random labels).*

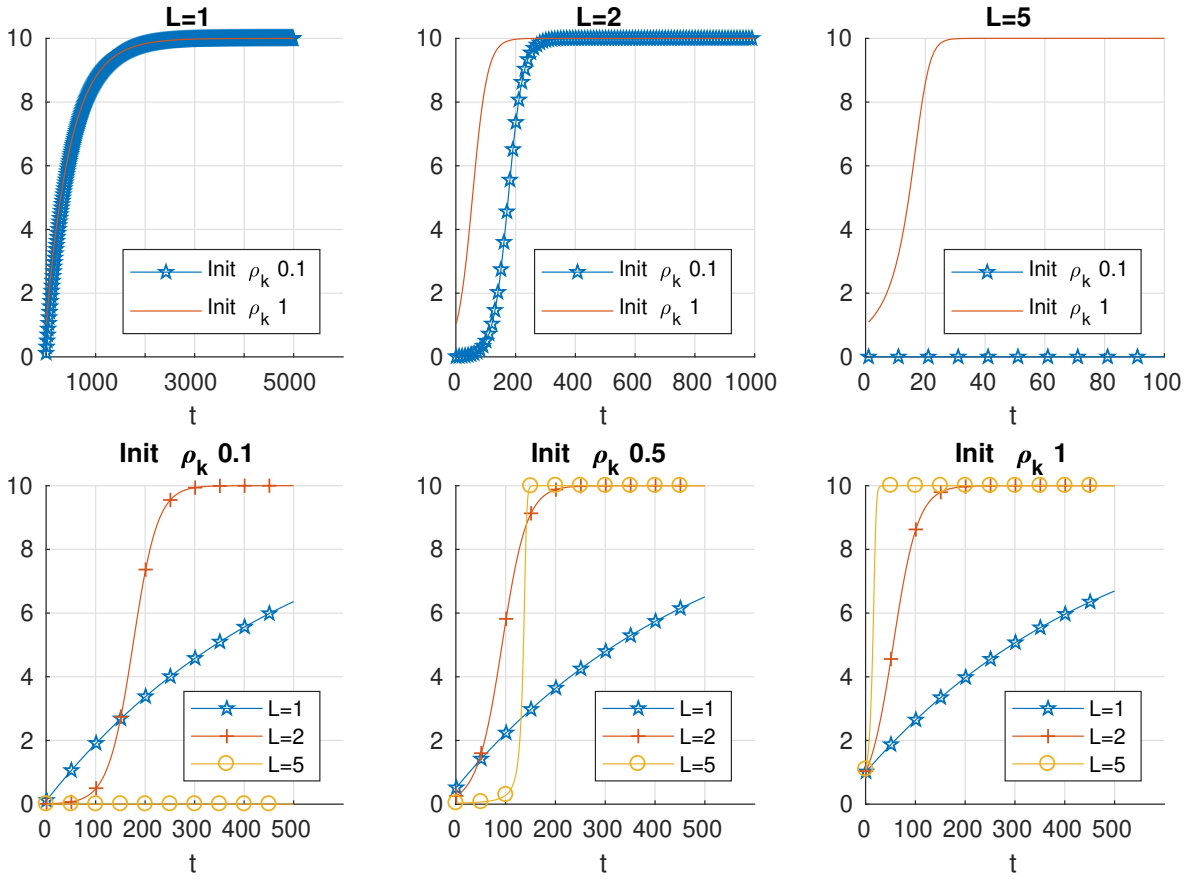


Figure 12: *Simulation of ρ from the logistic equation related to Equation* (57)*, in which the terms $\sum y_n f_n$ and $\sum f_n^2$ are positive constants.*

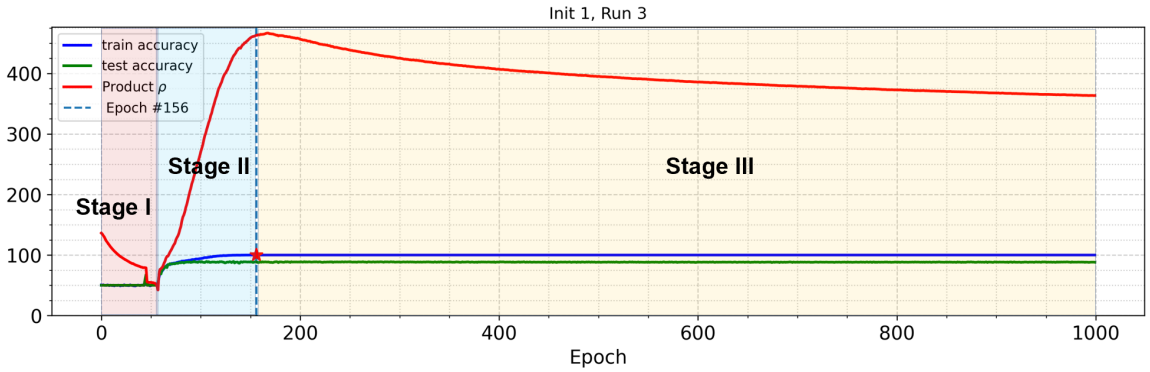


Figure 13: *Product ρ, training accuracy and test accuracy during model training with LN, initialization 1, weight decay and square loss. It shows three obvious stages during LN model training. Stage I: the red $\rho = \prod_k \rho_k$ curve decreased a bit (between epoch 0-57) when the training accuracy and test accuracy are 50% during this period); stage II: Total ρ starts to increase to some peak value at epoch 156 when the training accuracy will increase from 50% to 100%; Stage III: ρ start to decrease to some value with fast speed in the very beginning from epoch 100 to epoch 550, then slow down after epoch 550. Moreover, there is no significant changes in both train and test errors at stage III.*

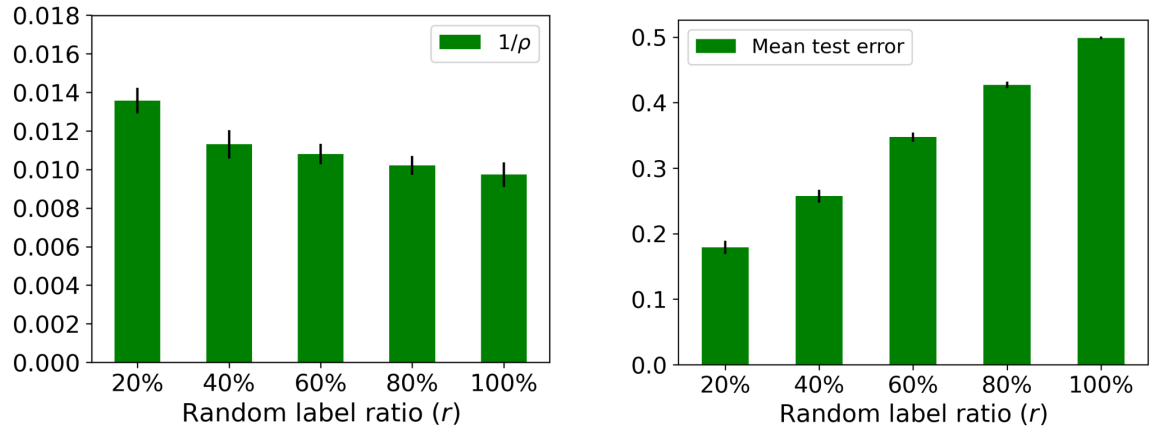


Figure 14: *Mean $1/\rho$ and test error results over 10 runs for binary classification on CIFAR10 trained with batch normalization and different percentages of random labels (r = 20%, 40%, 60%, 80% and 100%), initialization scale 0.1 and weight decay 0.01.*

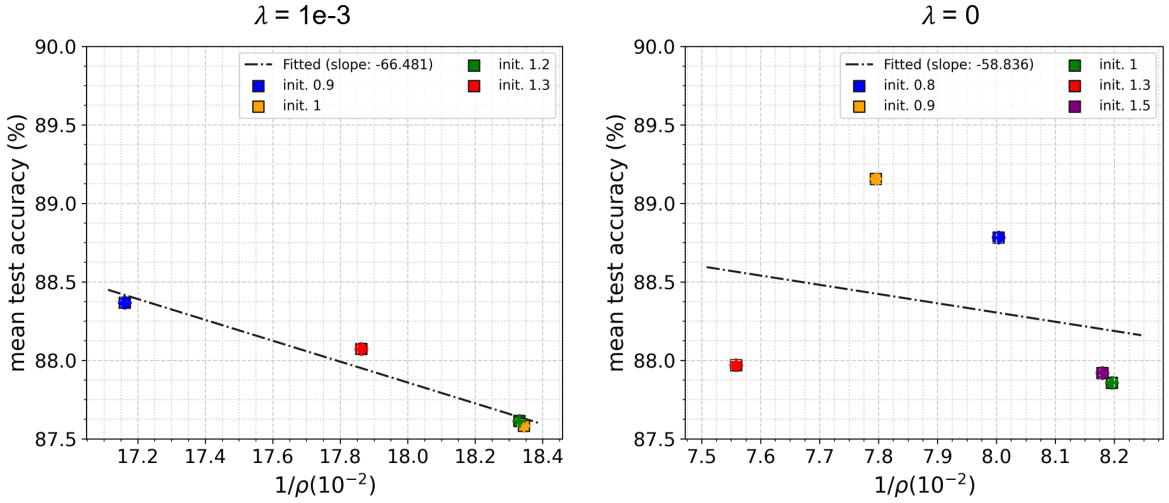


Figure 15: *Scatter plot for $1/\rho$ and mean test accuracy based on single run for binary classification on CIFAR10 using Lagrange Multiplier normalization, cross-entropy loss and Weight Decay (left) and without Weight Decay (right). In the left figure, the network was trained with different initialization scales (init. = [0.9, 1, 1.2, 1.3]) and with weight decay ($\lambda = 1e-3$); In the right figure, the network was trained with init. = [0.8, 0.9, 1, 1.3, 1.5] and no weight decay ($\lambda = 0$). When $\lambda > 0$, the coefficient (R^2), p-value and slope for linear regression between $1/\rho$ and mean test accuracy are: R^2 = 0.953, p-value = 0.024, slope = -66.481; When $\lambda = 0$, the coefficient R^2 = 0.072, p-value = 0.663 and the slope = -58.836.*

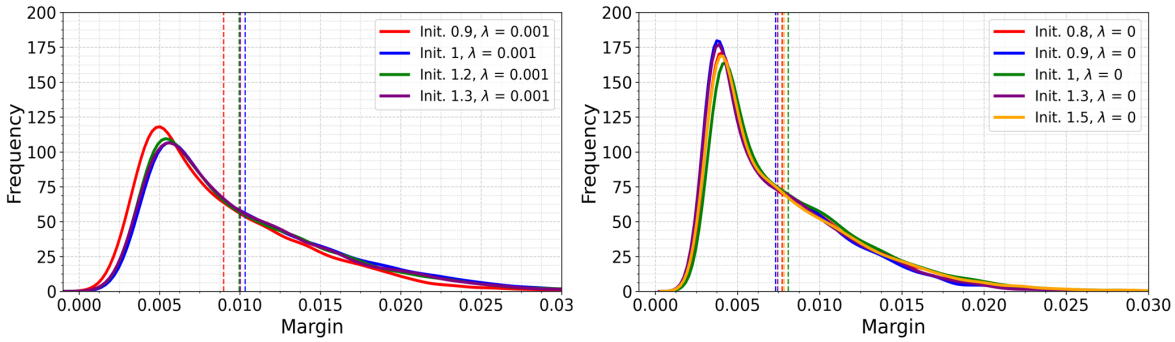


Figure 16: *Training margins for binary classification on the CIFAR10 dataset trained with cross-entropy loss, Lagrange Multiplier normalization and Weight Decay* $(\lambda) = 0.001$ $(left)$ *and without Weight Decay* $(right, \lambda = 0)$ *for different initializations* $(init. = 0.8, 0.9, 1, 1.2, 1.3$ *and* $1.5)$*. We applied a cosine learning rate scheduler with initial learning rate 0.01 during training. In the absence of weight decay* $(\lambda = 0)$ *there is clear evidence of SGD noise unlike in the square loss case.*