CENTER FOR
Brains
Minds+
Machines

CBMM Memo No. 90 **April 10, 2020**

Theory III: Dynamics and Generalization in Deep Networks[1]

Andrzej Banburski [1], Qianli Liao[1], Brando Miranda[1], Tomaso Poggio[1], Lorenzo Rosasco[1], Fernanda De La Torre[1], Jack Hidary[2]

[1]Center for Brains, Minds, and Machines, MIT
[2]Alphabet (Google) X

Abstract

The key to generalization is controlling the complexity of the network. However, there is no obvious control of complexity – such as an explicit regularization term – in the training of deep networks for classification. We will show that a classical form of norm control – but kind of hidden – is present in deep networks trained with gradient descent techniques on exponential-type losses. In particular, gradient descent induces a dynamics of the normalized weights which converge for $t \to \infty$ to an equilibrium which corresponds to a minimum norm (or maximum margin) solution. For sufficiently large but finite ρ – and thus finite t – the dynamics converges to one of several margin maximizers, with the margin monotonically increasing towards a limit stationary point of the flow. In the usual case of stochastic gradient descent, most of the stationary points are likely to be convex minima corresponding to a constrained minimizer – the network with normalized weights– which corresponds to vanishing regularization. The solution has zero generalization gap, for fixed architecture, asymptotically for $N \to \infty$, where N is the number of training examples. Our approach extends some of the original results of Srebro from linear networks to deep networks and provides a new perspective on the implicit bias of gradient descent. We believe that the elusive complexity control we describe is responsible for the puzzling empirical finding of good predictive performance by deep networks, despite overparametrization.

This material is based upon work supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216.

[1]This replaces previous versions of Theory IIIa and TheoryIIIb.

Theory III: Dynamics and Generalization in Deep Networks*

Andrzej Banburski , Qianli Liao, Brando Miranda, Tomaso Poggio,

Lorenzo Rosasco, Fernanda De La Torre, and Jack Hidary

Alphabet (Google) X

Abstract

The key to generalization is controlling the complexity of the network. However, there is no obvious control of complexity – such as an explicit regularization term – in the training of deep networks for classification. We will show that a classical form of norm control – but kind of hidden – is present in deep networks trained with gradient descent techniques on exponential-type losses. In particular, gradient descent induces a dynamics of the normalized weights which converge for $t \to \infty$ to an equilibrium which corresponds to a minimum norm (or maximum margin) solution. For sufficiently large but finite ρ – and thus finite t – the dynamics converges to one of several margin maximizers, with the margin monotonically increasing towards a limit stationary point of the flow. In the usual case of stochastic gradient descent, most of the stationary points are likely to be convex minima corresponding to a constrained minimizer – the network with normalized weights– which corresponds to vanishing regularization. The solution has zero generalization gap, for fixed architecture, asymptotically for $N \to \infty$, where N is the number of training examples. Our approach extends some of the original results of Srebro from linear networks to deep networks and provides a new perspective on the implicit bias of gradient descent. We believe that the elusive complexity control we describe is responsible for the puzzling empirical finding of good predictive performance by deep networks, despite overparametrization.

1 Introduction

In the last few years, deep learning has been tremendously successful in many important applications of machine learning. However, our theoretical understanding of deep learning, and thus the ability of developing principled improvements, has lagged behind. A satisfactory theoretical characterization of deep learning is emerging. It covers the following questions that are natural in machine learning techniques based on empirical risk minimization (see for instance [1], [2]: 1) *representation power* of deep networks 2) *optimization* of the empirical risk

*This replaces previous versions of Theory III, that appeared on the CBMM site and (much more sparsely) on arXiv.

3) *generalization properties* of gradient descent techniques — why the expected error does not suffer, despite the absence of explicit regularization, when the networks are overparametrized? We refer to the latter as the non-overfitting puzzle, around which several recent papers revolve (see among others [3, 4, 5, 6, 7]). This paper addresses the third question.

We start with recent observations on the dynamical systems induced by gradient descent methods used for training deep networks and summarize properties of the solutions they converge to. Remarkable results by [8] illuminate the apparent absence of "overfitting" in the case of linear networks trained on the exponential loss for binary classification. They prove that minimization of loss functions such as the logistic, the cross-entropy and the exponential loss yields asymptotic convergence to the maximum margin solution for linearly separable datasets, independently of the initial conditions and without explicit regularization. In this paper, we discuss the case of nonlinear multilayer DNNs in the setting of separable data, under exponential-type losses and square loss, for several variations of the basic gradient descent algorithm. Because of homogeneity of the RELU, deep networks can be represented as $f(W;x) = \rho f(V;x)$ where ρ is the product of the norms of the weight matrix W at each layer and $f(V;x)$ is the network with normalized weights V_k at layer k.

Our *main result* is that *unconstrained gradient descent* over an exponential-type loss – this is the usual training procedure for deep networks – *converges to solutions V_k that for long but finite time generalize* because they are equivalent to constrained minimization or equivalently to regularization schemes (which are stable and thus generalize). At the limit, they converge to a minimum norm solution which is not stable, does not generalize but may perform well (in analogy with pseudoinverse). Other results are:

- Consider gradient descent algorithms – such as Lagrange multipliers methods – that minimize the exponential loss while enforcing a unit L_p norm constraint on the normalized weights. The assumption that separability is reached during gradient descent implies convergence of the dynamics of the V_k to a limit stationary point of the flow for any finite, fixed ρ.

- For $\rho \to \infty$ the stationary points coincide with the stationary points of the full dynamical system obtained from the Lagrange multiplier formulation by minimizing also on V_k *and* ρ .

- These limit stationary points are minimum norm – and maximum margin – solutions.

- Standard gradient descent used in training deep networks in the unnormalized weights followed by L_2 normalization (performed after stopping gradient descent) has the same qualitative dynamics as the Lagrange method and has the same stationary points.

- Weight normalization and batch normalization have a similar qualitative dynamics and converge to the same stationary points.

In the perspective of these theoretical results, we discuss experimental evidence around the apparent absence of "overfitting", that is the observation that the expected classification error does not get worse when increasing the number of parameters.

2 Deep networks: definitions and properties

Definitions We define a deep network with K layers with the usual coordinate-wise scalar activation functions $\sigma(z) : \quad \mathbf{R} \to \mathbf{R}$ as the set of functions $f(W;x) = W^K \sigma(W^{K-1} \cdots \sigma(W^1 x))$, where the input is $x \in \mathbf{R}^d$, the weights are given by the matrices W^k, one per layer, with matching dimensions. We use the symbol W as a shorthand for the set of W^k matrices $k = 1, \cdots, K$. For simplicity we consider here the case of binary classification in which f takes scalar values, implying that the last layer matrix W^K is $W^K \in \mathbf{R}^{1,K_l}$. The labels are $y_n \in \{-1, 1\}$. The weights of hidden layer l are collected in a matrix of size $h_l \times h_{l-1}$. There are no biases apart form the input layer where the bias is instantiated by one of the input dimensions being a constant. The activation function in this paper is the ReLU activation. The norm we use is the L_2 unless we say otherwise.

Network homogeneity For ReLU activations the following positive one-homogeneity property holds $\sigma(z) = \frac{\partial \sigma(z)}{\partial z} z$. For the network this implies $f(W;x) = \prod_{k=1}^K \rho_k f(V_1, \cdots, V_K; x_n)$, where $W_k = \rho_k V_k$ with the matrix norm $||V_k||_p = 1$. This implies the following property of ReLU networks w.r.t. their Rademacher complexity:

$$\mathbb{R}_N(\mathbb{F}) = \rho \mathbb{R}_N(\tilde{\mathbb{F}}), \tag{1}$$

where $\rho = \rho_1 \cdots \rho_K$, $\mathbb{F}$ is the class of neural networks described above and accordingly $\tilde{\mathbb{F}}$ is the corresponding class of normalized neural networks $f(V;x)$[1]. In the paper we will refer to the product $\rho = \prod_{k=1}^K \rho_k$ of the norms of the K weight matrices of f. Note that

$$\frac{\partial f(W;x)}{\partial \rho_k} = \frac{\rho}{\rho_k} f(V;x) \tag{2}$$

and that the definitions of ρ_k, V_k and $f(V;x)$ all depend on the choice of the norm used in normalization.

Structural property The following structural property of the gradient of deep ReLU networks is useful (Lemma 2.1 of [9]):

$$\sum_{i,j} \frac{\partial f(W;x)}{\partial W_k^{i,j}} W_k^{i,j} = f(W;x); \tag{3}$$

for $k = 1, \cdot, K$. Equation 3 can be rewritten as an inner product between W_k *as vectors !*:

$$(W_k, \frac{\partial f(W;x)}{\partial W_k}) = f(W;x) \tag{4}$$

where W_k is here the vectorized representation of the weight matrices W_k for each of the different layers[2]. The same property holds for V_k

[1] This invariance property of the function f under transformations of W_k that leaves the product norm the same is typical of ReLU (and linear) networks.

[2] By taking derivatives of both sides of Equation 4, we obtain to the following property

$$(V_k, \frac{\partial f(V;x)}{\partial V_k}) = f(V;x) \tag{6}$$

Gradient flow The gradient flow of the empirical risk L is often written as

$$\dot{W} \equiv \frac{dW}{dt} = -\gamma(t)\nabla_W(L(f)), \tag{7}$$

where $\gamma(t)$ is the learning rate (in this paper we will neglect it). We are well aware that the continuous formulation and the discrete one are not equivalent but we are happy to leave a careful analysis – especially of the discrete case – to better mathematicians.

We conjecture that the hypothesis of smooth activations is just a technicality due to the necessary conditions for existence and uniqueness of solutions to ODEs. Generalizing to differential inclusions and non-smooth dynamical systems should allows for these conditions to be satisfied in the Filippov sense [10]. The Clarke subdifferential is supposed to deal appropriately with functions such as RELU.

Separability When $y_n f(W;x_n) > 0 \ \forall n = 1, \cdots, N$ we say that the data are separable wrt $f \in \mathbf{F}$, that is they can all be correctly classified. We assume in this paper that there exist T_0 such that for $t > T_0$ gradient descent attains a f that separates the data. Notice that this is a strong condition on the data if f is linear but it is a weak assumption in the case of overparametrized, nonlinear, deep networks. In fact here we assume that the condition of *separability is reached during gradient descent* by the networks we consider.

3 Related work

There are many recent papers studying optimization and generalization in deep learning. For optimization we mention work based on the idea that noisy gradient descent [11, 12, 13, 14] can find a global minimum. More recently, several authors studied the dynamics of gradient descent for deep networks with assumptions about the input distribution or on how the labels are generated. They obtain global convergence for some shallow neural networks [15, 16, 17, 18, 19, 20]. Some local convergence results have also been proved [21, 22, 23]. The most interesting such approach is [20], which focuses on minimizing the training loss and proving that randomly initialized gradient descent can achieve zero training loss (see also [24, 25, 26]). In summary, there is by now an extensive literature on optimization that formalizes and refines to different special cases and to the discrete domain our results of Theory II and IIb (see section 6).

$$(W_k, \frac{\partial^2 f(W;x)}{\partial {W_k}^2}) = 0. \tag{5}$$

From Equation 3, it follows that the condition $\left(\frac{\partial f(W;x)}{\partial W_k}\right) = 0$ implies $f(x) = 0$. In the case of square loss, this condition restricts the non-fitting stationary points of the gradient to be either linear combinations $\sum_{n=1}^{N}(f(W;x_n) - y_n)\frac{\partial f(W;x)}{\partial W_k} = 0$, with $\frac{\partial f(W;x)}{\partial W_k} \neq 0, \forall k$ or $f(W;x) = 0$. A similar restriction also holds for the exponential loss (see Equation 8).

For generalization, existing work demonstrate that gradient descent works under the same situations as kernel methods and random feature methods [27, 28, 29]. Closest to our approach – which is focused on the role of batch and weight normalization – is the paper [30]. Its authors study generalization assuming a regularizer because they are – like us – interested in normalized margin. Unlike their assumption of an explicit regularization, we show here that commonly used techniques, such as batch normalization as well as weight normalization, maximize margin while controlling the complexity of the classifier without the need to add a regularizer or to use weight decay. In fact, we will show that even standard gradient descent on the weights controls the complexity of the normalized weights.

Very recently, well after previous versions of this work, two papers ([31] and [32]) appeared. They develop an elegant but complicated margin maximization based approach, describing the relations between the *margin, the constrained and the optimization paths* deriving some of the same results of this section (and more). Our approach does not need the notion of maximum margin but our theorem 7 establishes a connection with it and thus with the results of [31] and [32]. Our main focus here (and in [33]) is on the puzzle of how complexity of deep nets is controlled during training despite overparametrization and despite the absence of regularization. Our main original contribution is a study of the *gradient flow of the normalized weights* to characterize the implicit *control responsible for generalization* in deep networks trained under the exponential loss, which describe as implicit L_2 normalization by gradient descent. In the process of doing this, we analyze the dynamics of the flow of *the direction of the weights* induced by gradient descent on the unnormalized weights.

4 Main results

The standard approach to training deep networks is to use stochastic gradient descent to find the weights W_k that minimize the empirical exponential loss $L = \sum_n e^{-y_n f(W;x_n)}$ by computing

$$\dot{W_k} = -\frac{\partial L}{\partial W_k} = \sum_{n=1}^{N} y_n \frac{\partial f(W;x_n)}{\partial W_k} e^{-y_n f(W;x_n)} \tag{8}$$

on a given dataset $\{x_i, y_i\} \quad \forall i = 1, \ldots, N$.

In this section we study three related versions of this problem:

1. the minimization of $L = \sum_n e^{-\rho y_n f(V;x_n)}$ under the constraint $||V_k|| = 1$ wrt V_k for fixed ρ;
2. the minimization of $L = \sum_n e^{-\rho y_n f(V;x_n)}$ under the constraint $||V_k|| = 1$ wrt V_k, ρ;
3. the minimization of $L = \sum_n e^{-\rho y_n f(V;x_n)} = \sum_n e^{-y_n f(W;x_n)}$ wrt V_k, ρ, which is the standard situation for deep nets.

The section is organized as follows. We will show that problem 1) above converges to a stationary point for any finite ρ and then that for $\rho \to \infty$ the stationary points of the flow of 1) are the same as the minima of 2) for $t \to \infty$. Asymptotically these stationary points are margin

maximizers and correspond (see Appendix 9) to minimum norm solutions. We will then prove that the gradient descent system associated with 3) has the same stationary equilibria as problem 2). Finally, we observe that for any finite t the solution is regularized, as in early stopping for linear networks regression [34] (for $(\lambda(\rho(t))N)^{-\frac{1}{2}}\ \epsilon$). ; for $t \to \infty$ the solutions are minimum norm minimizers, a situation similar to the case of the pseudoiverse for the linear regression case [34].

4.1 Constrained minimization of the exponential loss

Generalization bounds suggest constrained optimization of the exponential loss that is to minimize $L = \sum_n e^{-\rho y_n f(V;x_n)}$ under the constraint $||V_k|| = 1$ which leads to minimize

$$\mathcal{L} = \sum_n e^{-\rho y_n f(V;x_n)} + \sum_k \lambda_k ||V_k||^2 \tag{9}$$

with λ_k such that the constraint $||V_k|| = 1$ is satisfied.

4.2 Fixed ρ: stationary points of the gradient flow

Gradient descent on $\mathcal{L}$ for fixed ρ wrt V_k yields the dynamical system

$$\dot{V_k} = \rho \sum_n e^{-\rho y_n f(V;x_n)} y_n (\frac{\partial f(V;x_n)}{\partial V_k} - V_k f(V;x_n)) \tag{10}$$

because $\lambda_k = \frac{1}{2}\rho \sum_n e^{-\rho f_V(x_n)} f_V(x_n)$, since $\sum_{i,j} (V_k)^{i,j} (\dot{V_k})_{i,j} = 0$ because $||V_k||^2 = 1$.

To shorten the expressions that will appear multiple times, let us define

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

so that the dynamical system becomes

$$\dot{V_k} = \rho \sum_n e^{-\rho y_n f(V;x_n)} y_n S_k \frac{\partial f(V;x_n)}{\partial V_k}.$$

Let us consider $L(V_k) = \sum_n e^{-\rho y_n f(V;x_n)}$ assuming that (after convergence of GD) $||V_k|| = 1$. Since for fixed ρ the domain is compact, minima and maxima $\frac{\partial L}{\partial V_K} = 0$ of the constrained optimization problem must exist for each fixed ρ. Assuming data separation is achieved, they satisfy

$$\sum_n e^{-\rho y_n f(V;x_n)} \frac{\partial f(V;x_n)}{\partial V_k} = \sum_n e^{-\rho y_n f(V;x_n)} V_k f(V;x_n) \tag{11}$$

Of course the minimum of the exponential loss L is only zero for the limit $\rho = \infty$; for any finite ρ the minimum of L is at the boundary of the compact domain. For any finite, sufficiently large ρ the minimum is critical point of the gradient with a positive semidefinite Hessian (see Appendix 13.0.1). In general it is not unique; it is unique in the case of one hidden layer linear networks.

4.2.1 Detailed Analysis

For ρ sufficiently large, choose the two x_n for which $f(V; x_n)$ is smallest. Let us call them x_1 and x_2 with $f(x_1) < f(x_2)$. The assumption of letting ρ increase to infinity is in practice equivalent to set ρ to a value such that $e^{-\rho y_n f(V;x_2)}$ corresponds to zero at machine precision (whatever it is). For this value of ρ then the stationary point satisfies

$$\frac{\partial f(V; x_1)}{\partial V_k} = V_k f(V; x_1). \tag{12}$$

What if the dynamical system is run with a smaller, fixed ρ than the "infinity" ρ assumed above? In this case, there may be or not

4.3 $\rho \to \infty$ has same stationary points as the full dynamical system

Consider the limit of $\rho \to \infty$ in Equation 11 . The asymptotic stationary points of the flow of V_k then satisfy

$$\sum_n e^{-\rho y_n f(V;x_n)} y_n S_k \frac{\partial f_V(x_n)}{\partial V_k} = 0 \tag{13}$$

also in the limit $\lim_{\rho\to\infty}$, that is for any large ρ. So the stationary V_k points for any large $\rho = R$ satisfy

$$\sum_n e^{-R y_n f(V;x_n)} y_n (\frac{\partial f(V; x_n)}{\partial V_k} - V_k f(V; x_n)) = 0. \tag{14}$$

Consider now gradient descent for the full system obtained with Lagrange multipliers, that is, on $\mathcal{L} = \sum_n e^{-\rho y_n f(V;x_n)} + \sum_k \lambda_k ||V_k||^2$ wrt V_k *and* ρ_k, with λ_k chosen (as before) to implement the unit norm constraint. The full gradient dynamical system is

$$\begin{aligned} \dot{\rho_k} &= \frac{\rho}{\rho_k} \sum_n e^{-\rho y_n f(V;x_n)} y_n f(V; x_n) \\ \dot{V_k} &= \rho \sum_n e^{-\rho y_n f(V;x_n)} y_n S_k \frac{\partial f(V; x_n)}{\partial V_k} \end{aligned} \tag{15}$$

Observe (see Appendix) that after onset of separability $\dot{\rho_k} > 0$ with $\lim_{t\to\infty} \dot{\rho_k} = 0$, $\lim_{\to\infty} \rho(t) = \infty$ (for one layer $\rho \propto \log t$; for more layer it is faster, see Appendix 12). Appendix 12 shows that $\rho(t)$ is a monotonically increasing function from $t = 0$ to $t = \infty$ and that ρ_k grow at the same rate, independently of the layer k. Thus for any large R in Equation 14, there exist T such that $\rho(T) = R$. At time T then, the condition for the stationary point of the V_k in Equation 15 is

$$\dot{V_k} = R \sum_n e^{-R y_n f(V;x_n)} y_n S_k \frac{\partial f(V; x_n)}{\partial V_k} = 0 \tag{16}$$

which coincides exactly with Equation 13.

Thus the full dynamical system 15 in the limit of $t \to \infty$ converges to the same limit – if it exists – as does the dynamical system Equation 10 for $\rho \to \infty$ (in the sense that they have the same stationary points).

4.4 Asymptotic stationary points coincide with minimal norm (maximum margin) minimizers

Here we show that the limit for the two systems exists, is not trivial and corresponds to maximum margin/minimum norm solutions. We start from Equation 13. Without loss of generality let us assume that the training data $f(V; x_n)$ are ranked at $t = T_0$ according to increasing normalized margin, that is $f(V; x_1) \leq f(V; x_2) \leq \cdots \leq f(V; x_N)$. Let us call $B_k(V; x_n) = y_n(\frac{\partial f(V;x_n)}{\partial V_k} - V_k f(V; x_n))$. Then the equilibrium condition of Equation 13 becomes

$$\begin{aligned} &\sum_n^N e^{-\rho(T_0) y_n f(V;x_n)} B_k(V; x_n) = \\ &e^{-R y_1 f(V;x_1)}(B_1 + B_2 e^{-R\Delta_2} + \cdots + B_N e^{-R\Delta_N}) = 0 \end{aligned} \tag{17}$$

where $\Delta_1 = y_1 f(V; x_1) - y_1 f(V; x_1) = 0$, $\Delta_2 = y_2 f(V; x_n) - y_1 f(V; x_1) \geq 0$, $\Delta_3 = y_3 f(V; x_n) - y_1 f(V; x_1) \geq 0$ and all $\Delta_n \geq 0$ increase with n. The equation can be rewritten then as

$$e^{-\rho(T_0) y_1 f(V;x_1)}(\alpha_1 B_1 + \alpha_2 B_2 + \cdots + \alpha_N B_N) = 0 \tag{18}$$

where the α_n are all positive ($\alpha_1 = 1$) and decreasing with n.

The left hand side of the stationary point equation has the form $\epsilon(B_1 + \epsilon' B) = 0$, with $B = \sum_{n=2}^N \alpha_n B_n$, with both ϵ and ϵ' going to zero for $T_0 \to \infty$. Thus $\lim_{T_0 \to \infty} \epsilon(B_1 + \epsilon' B) = \lim_{T_0 \to \infty} \epsilon B_1 = 0$ can be satisfied for sufficiently large T_0 by $B_1 = 0$, that is by $\frac{\partial f(V;x_1)}{\partial V_k} - V_k f(V; x_1) = 0$. This is the condition in which the stationary point corresponding to x_1 provides the maximum margin. Before that limit is reached, the solution V_k changes with increasing $\rho(t)$. Thus the asymptotic stationary points coincide with maximum margin. The following lemma [33] shows that the margin is increasing with t, after separability is reached and after a single support vector dominates:

Lemma 1 *There exists a $\rho(T_*)$ such that for $\rho > \rho(T_*)$ the sum $\sum_{n=1}^N e^{-\rho y_n f(V;x_n)} \propto e^{-\rho y_1 f(V;x_1)}$. For $\rho > \rho(T_*)$ then the margin increases $y_1 \frac{\partial f(V;x_1)}{\partial t} \geq 0$ (even if ρ is kept fixed).*

Proof $y_1 f(V; x_1)$ increases monotonically or is constant because

$$\begin{aligned} &y_* \frac{\partial f(V; x_1)}{\partial t} = \sum_k (\frac{\partial y_1 f(V; x_1)}{\partial V_k})^T \dot{V}_k = \\ &\sum_k \frac{\rho}{\rho_k^2} e^{-\rho y_1 f(V;x_1)} \left(\left\| \frac{\partial f(V; x_1)}{\partial V_k} \right\|_F^2 - f(V; x_1)^2 \right). \end{aligned} \tag{19}$$

Equation 75 implies $\frac{\partial y_1 f(V,x_1)}{\partial t} \geq 0$ because $||f(V;x_1)||_F = ||V_k^T \frac{\partial f(V;x_1)}{\partial V_k}||_F \leq ||\frac{\partial f(V;x_1)}{\partial V_k}||_F$, since the Frobenius norm is sub-multiplicative and V_k has unit norm.

We finally note that the maximum margin solution in terms of $f(V;x)$ and V_k is equivalent to a minimum norm solution in terms of W_k under the condition of the margin being at least 1. This is stated in the following lemma (see Appendix):

Lemma 2

The maximum margin problem

$$\max_{W_K,\cdots,W_1} \min_n y_n f(W;x_n), \quad \textit{subj. to} \quad \|W_k\| = 1, \quad \forall k. \tag{20}$$

is equivalent to

$$\min_{W_k} \frac{1}{2}\|W_k\|^2, \ \textit{subj. to } y_n f(W;x_n) \geq 1, \quad \forall k, \ n = 1,\ldots,N. \tag{21}$$

4.5 Unconstrained gradient descent

Empirically it appears that GD and SGD converge to solutions that can generalize even without any explicit capacity control such as a regularization term or a constraint on the norm of the weights. How is this possible? The answer is provided by the fact – trivial or surprising – that the unit vector $\frac{w(T)}{||w(T)||_2}$ computed from the solution $w(T)$ of gradient descent $\dot{w} = -\nabla_w L$ at time T is the same, irrespectively of whether the constraint $||v||_2 = 1$ is enforced during gradient descent. This confirms Srebro results for linear networks, extending some of them to the deep network case. It also throws some light on the nature of the implicit bias or hidden complexity control. We show this result next.

4.5.1 Reparametrization of standard gradient descent

We study the new dynamical system induced by the dynamical system in $\dot{W_k}^{i,j}$ under the reparametrization $W_k^{i,j} = \rho_k V_k^{i,j}$ with $||V_k||_2 = 1$. This is equivalent to changing coordinates from W_k to V_k and $\rho_k = ||W_k||_2$. For simplicity of notation we consider here for each weight matrix V_k the corresponding "vectorized" representation in terms of vectors $W_k^{i,j} = W_k$.

We use the following definitions and properties (for a vector w):

- The norm $||\cdot||$ is assumed in this section to be the L_2 norm.
- Define $\frac{w}{\rho} = v$; thus $w = \rho v$ with $||v||_2 = 1$ and $\rho = ||w||_2$.
- The following relations are easy to check:
  1. $\frac{\partial ||w||_2}{\partial w} = v$
  2. Define $S = I - vv^T = I - \frac{ww^T}{||w||_2^2}$. S has at most one zero eigenvalue since vv^T is rank 1 with a single eigenvalue $\lambda = 1$. This means also $S \geq 0$, as can be seen directly.

3. $\frac{\partial v}{\partial w} = \frac{S}{\rho}$.
4. $Sw = Sv = 0$
5. $S^2 = S$
6. In the multilayer case, $\frac{\partial f(x_n;W)}{\partial W_k} = \frac{\rho}{\rho_k}\frac{\partial \tilde{f}(V;x_n)}{\partial V_k}$

The unconstrained gradient descent dynamic system used in training deep networks for the exponential loss is given in Equation 8, that is

$$\dot{W}_k = -\frac{\partial L}{\partial W k} = \sum_{n=1}^{N} y_n \frac{\partial f(W;x_n)}{\partial W_k} e^{-y_n f(W;x_n)}. \tag{22}$$

Following the chain rule *for the time derivatives*, the dynamics for W_k induces the following dynamics for $||W_k|| = \rho_k$ and V_k:

$$\dot{\rho_k} = \frac{\partial ||W_k||}{\partial W_k}\frac{\partial W_k}{\partial t} = V_k^T \dot{W}_k \tag{23}$$

and

$$\dot{V_k} = \frac{\partial V_k}{\partial W_k}\frac{\partial W_k}{\partial t} = \frac{S_k}{\rho_k}\dot{W}_k \tag{24}$$

where $S_k = I - V_k V_k^T$. We now obtain the time derivatives of V_k and ρ_k from the time derivative of W_k; the latter is computed from the gradients of L with respect to W_k that is from the gradient dynamics of W_k. Thus *unconstrained gradient descent* coincides with the following dynamical system

$$\dot{\rho_k} = V_k^T \dot{W}_k = \sum_{n=1}^{N} V_k^T y_n \frac{\partial f(W;x_n)}{\partial W_k} e^{-y_n f(W;x_n)} = \frac{\rho}{\rho_k}\sum_{n=1}^{N} y_n e^{-\rho y_n f(V;x_n)} y_n f(V;x_n) \tag{25}$$

and

$$\dot{V_k} = \frac{\rho}{\rho_k^2}\sum_{n=1}^{N} e^{-\rho y_n f(V;x_n)} y_n \left(\frac{\partial f(V;x_n)}{\partial V_k} - V_k f(V;x_n)\right), \tag{26}$$

where we used the structural lemma to set $V_k V_k^T \frac{\partial \tilde{f}(x_n)}{\partial V_k} = V_k \tilde{f}(x_n)$.

Clearly the dynamics of *unconstrained gradient descent* and the dynamics of *constrained gradient descent* are very similar since they differ by a ρ^2 factor in the $\dot{v}$ equations. The conditions for the stationary points of the gradient for the v vectors – that is the values for which $\dot{v} = 0$ – are *the same in both cases* since for any $t > 0$ $\rho(t) > 0$.

4.5.2 Constrained optimization and weight normalization

We recall that *constrained gradient descent* using Lagrange multipliers yields the dynamical system

$$\begin{aligned}\dot{\rho_k} &= \frac{\rho}{\rho_k}\sum_n e^{-\rho y_n f(V;x_n)} y_n f(V;x_n) \\ \dot{V_k} &= \rho\sum_n y_n e^{-\rho y_n f(V;x_n)}(\frac{\partial f(V;x_n)}{\partial V_k} - V_k f(V;x_n)).\end{aligned} \tag{27}$$

Constrained normalization by tangent gradient gives the same dynamical system, as expected. The Appendix shows that it coincides with the so-called weight normalization algorithm.

4.6 Linear networks

In the case of linear networks under the exponential loss, we use stochastic gradient descent to find the weights w that minimize the empirical exponential loss $L = \sum_n e^{-y_n w^T x_n}$ by computing

$$\dot{w} = -\frac{\partial L}{\partial w} = \sum_{n=1}^{N} y_n x_n e^{-y_n w^T x_n} \tag{28}$$

We assume linear separability, that is $w^T x_n y_n > 0$. The dynamical system 28 diverges with $||w|| \to \infty$. We consider instead the gradient dynamics of the system with Lagrange multipliers and $L = \sum_n e^{-\rho y_n v^T x_n} + \lambda(||v||^2 - 1)$

$$\dot{\rho} = \sum_{n=1}^{N} e^{-\rho y_n v^T x_n} y_n v^T x_n \tag{29}$$

and

$$\dot{v} = -\frac{\partial L}{\partial v} = \rho\sum_{n=1}^{N} e^{-\rho y_n v^T x_n} y_n x_n - 2\lambda v. \tag{30}$$

The value of λ that ensures $||v|| = 1$ is $\lambda = \frac{\rho}{2}\sum_n e^{-\rho y_n v^T x_n} y_n v^T x_n$. The critical points correspond to $\dot{v} = 0$ that is to $\rho \to \infty$ or if a set of support vectors with value $f(w;x^*)$ is reached at a finite time to

$$x^* - vv^T x^* = 0 \tag{31}$$

which gives $v = x^*$ for the value of the weights at the critical point which has to be a minimum of the loss and a maximum of the margin (see earlier).

The Hessian of L (and Jacobian of L) is given by

$$H^{ij} = \sum_n e^{-\rho y_n v^T x_n}\rho^2 x_n^i x_n^j + 2\lambda\delta^{ij}.$$

If we now plug in the value of λ, we get

$$H^{ij} = \sum_n e^{-\rho y_n v^T x_n} (\rho^2 x_n^i x_n^j + \rho y_n v^T x_n \delta^{ij})$$

Since $x_n x_n^T$ is positive semi-definite, and we have separability, the Hessian of the loss is positive definite and, in particular, it is positive definite at the critical point $v = x^*$.

4.7 Main result

Actual convergence for the constrained case at a finite time happens in the case of a set of support vectors with the same margin $\sum_n^N e^{-Ry_n f(V;x_n)} y_n (\frac{\partial f(V;x_n)}{\partial V_k} - V_k f(V;x_n)) = (\frac{\partial f(V;x_*)}{\partial V_k} - V_k f(V;x_*)) = 0$ is valid, which corresponds to a large but finite ρ and a small but non-zero λ. The unconstrained case has the same solutions. There are obvious limitations to this asymptotic statement. In fact, it turns out that the dynamics we described converges for a certain N to a minimum norm, maximum margin solution similar to a pseudoinverse which does not technically generalize (expected error equal to empirical error) for that value of N but can perform well in terms of expected error. In addition, while an appropriately normalized network $f(V;x)$ can have a small generalization gap at some finite N under the exponential loss, bounds on the classification error for the same finite N remain an open problem. More importantly, the case $\sum_n^N e^{-Ry_n f(V;x_n)} = e^{-\rho y_* f(V;x_*)}$ may never happens during the finite times of a gradient descent run.

4.8 Summary

The following theorem summarizes our main results

Theorem 3 *Assume that separability is reached at time T_0 during gradient descent on the exponential loss, that is $y_n f(W;x_n) > 0, \quad \forall n$. Then unconstrained gradient descent converges in terms of the normalized weights to a solution that is under complexity control for any finite time. In addition the following properties hold:*

1. *Consider the dynamics (A) resulting from using Lagrange multipliers on the constrained optimization problem: "minimize $L = \sum_n e^{-\rho y_n f(V;x_n)}$ under the constraint $||V_k|| = 1$ wrt V_k". The dynamics converges for any fixed ρ to stationary points of the V_k flow that are minima with a positive semi-definite Hessian.*
2. *Consider the dynamics (B) resulting from using Lagrange multipliers on the constrained optimization problem: "minimize $L = \sum_n e^{-\rho y_n f(V;x_n)}$ under the constraint $||V_k|| = 1$ wrt V_k and ρ_k". The stationary points of V_k in (B) in the limit of $t \to \infty$ coincide with the limit $\rho \to \infty$ in the dynamics (A) and they are maxima of the margin.*
3. *The unconstrained gradient descent dynamics converges to the same stationary points of the flow of V_k as (A) and (B).*
4. *Weight normalization[35] corresponds to dynamics (B).*

5. *For each layer* $\frac{\partial \rho_k^2}{\partial t}$ *is the same irrespectively of* k.
6. *In the 1-layer network case* $\rho \approx \log t$ *asymptotically. For deeper networks, the product of weights at each layer diverges faster than logarithmically, but each individual layer diverges slower than in the 1-layer case*
7. *Gradient flow converges for infinite time to directions given by a set of support vectors with the same value* $y_1 f(V; x_1)$ *that satisfy the set of* K *coupled vector equations* ($\frac{\partial f(V;x_1)}{\partial V_k} = V_k f(V; x_1)$).

In the Appendix 16 we describe analysis of convergence for linear networks.

5 Discussion

Our main results are *for classification* in the setting of *separable data* for *continous gradient flow* under *the exponential loss.*

The key assumption is that at some point during gradient descent separability is reached. The assumption is reasonable because of overparametrization and empirically satisfied for many data sets. However, an analysis of the dynamical system before separability is lacking so far and should be the goal of further work. Our main result is that there is an *implicit* L_2 *norm constraint on the* V_k *dynamics in standard gradient descent for deep networks with RELUs.* Therefore standard gradient descent on the weights, provides a solution $f(V; x)$ that generalizes without the need of additional explicit regularization or explicit norm constraints.

To reach that conclusion we establish several properties of the dynamical system that are interesting in their own. As a starting point, the Appendix proves, under the assumption of separability, that for each layer $\frac{\partial \rho_k^2}{\partial t}$ grows monotonically with t, at the same rate irrespectively of k. In the 1-layer network case $\rho \approx \log t$ asymptotically. For deeper networks, the product of weights at each layer diverges faster than logarithmically, but each individual layer diverges slower than in the 1-layer case. Then, the dynamics of the normalized weights in *the standard unconstrained gradient descent* on the exponential loss has *the same asymptotic stationary points as gradient descent on the regularized loss (with vanishing* λ*).* Furthermore, the equilibrium point of the flow of V_k for $\rho \to \infty$ corresponds to a minimum norm, maximum margin solution of the constrained optimization problem.

Notice that gradient descent methods minimizing an empirical loss with a L_p norm constraint implement a classical recipe for good predictivity. They converge to stationary points of the gradient of V_k which attain zero loss – assuming separability occurs during gradient descent – for $\rho \to \infty$. The dynamics of W_k for separable data attain the global zero minimum of the loss for $||W_k|| \to \infty$ but diverges in W. The dynamics of the direction of the weights V_k in a multilayer network under SGD is likely to have several convex minima for any finite ρ which is sufficiently large.

Examples of techniques commonly used to train over-parametrized, multilayer, RELU, deep networks are *weight normalization* and *batch normalization* enforcing an explicit unit constraint

in the L_2 norm of V_k. For linear networks (and for one support vector) the convergence rate of standard GD ($\frac{1}{\log t}$) is slower than the convergence rate ($\frac{1}{t^{\log(\sqrt(t)}}$) of weight normalization.

The fact that the solution corresponds to a maximum margin (or minimum norm) solution may explain the following puzzling behavior (see Figures in the Supplementary Material, in which batch normalization was used): the test classification error does not get worse when the number of parameters increases well beyond the number of training data. This may be because the dynamical system is trying to maximize the *margin* under unit norm of $f(V; x)$.

The gradient flows corresponding to normalization with different L_p norms are expected to be different and to converge to different solutions, as in the classical case of support vector machines with L_2 vs L_1 regularizers. It is useful to emphasize that despite the similarities between some of the methods enforcing unit constraints in the 2-norm, they usually correspond to different dynamical flows but with the same qualitative dynamics and the same stationary solutions. In particular, batch normalization and weight normalization do not have the same dynamics; in turn they are slightly different from standard gradient descent. Furthermore, our analysis has been restricted to the continuous case; the discrete case may yield greater differences. An additional remark is that *weight normalization and batch normalization are enforcing explicitly a constraint (on the dynamics of V_k) which is natural for obtaining generalization*, and is deeper than reducing covariate shifts (the properties described in [36] are fully consistent with our characterization in terms of a norm constraint). Notice that as in the case of the vanishing regularizer assumed in the original theorem of [37], the *Lagrange multipliers λ_k should go to zero fast enough for $\lambda_k \rho_k$ to go to zero while $\rho_k \to \infty$.*

The basic complexity control mechanism we uncovered– regularization – explains the asymptotic generalization behavior (for $N \to \infty$) of deep networks but does not explain the more common regime of $N << D$ in which overparametrized deep networks fit the training data and perform well on out-of-sample points. It is therefore useful to recall that the classical analysis of Empirical Risk Minimization (ERM) algorithms studies their asymptotic behavior for the number of data N going to infinity. In this limiting regime, $N > D$ where D is the fixed number of weights; consistency (informally the expected error of the empirical minimizer converges to the best in the class) and generalization (the empirical error of the minimizer converges to the expected error of the minimizer) are equivalent. The capacity control described in this note implies that *there is asymptotic generalization and consistency in deep networks* for $n \to \infty$ and fixed architecture.

The non-asymptotic behavior in the overparametrized regime is similar to the regression case of linear kernel methods [38, 39, 40, 41]). This phenomenon suggests that under certain conditions, the pseudoinverse[3] may perform well in terms of expected error while the generalization gap (difference between expected and empirical loss) is large. We note that is not surprising that

[3]It is not well known, but easy to verify using the Mathlab function "cond", that the condition number associated with a random data matrix is usually worse for $N = D$, better for $N > D$ and even better for $N << D$. The proof is probably well-know to the experts since we found versions of it on the margins of a few linear algebra books. The implication is that data errors are amplified most when $N = D$ in the estimation of new data; in an equicalent way the ratio of out-of-sample prediction and in-sample-prediction is largest for $N = D$, better for $N > D$ and best for $N << D$.

complexity control is a prerequisite for good performance even in an overparametrized regime in which the classical analysis via generalization does not apply. The pseudoinverse solution is unique and continuous, satisfying the key conditions – including stability with respect to noise in the data – of a well-posed problem. We also remark that the analysis of the dynamics of deep networks described in this paper, once adapted to the square loss, suggests that the weights W_k of each layer converge to local minimum norm minimizers, because of the iterative regularization properties of gradient descent (in analogy with the fully linear case [34]).

In addition, commonly used weight decay with appropriate parameters can induce generalization since it is equivalent to regularization. Furthermore, typical implementations of data augmentation may also effectively avoid overparametrization: at each iteration of SGD only "new" data are used and depending on the number of iterations it is quite possible that the size of the training data exceeds the number of parameters. Within this online framework, one expects convergence to the minimum of the expected risk (see Supplementary Material section on Data Augmentation) without the need to invoke generalization bounds.

There are of course several open problems. It seems that under certain conditions neural networks can be described in terms of a "neural tangent kernel" in a linear way (wrt weights) – as hinted in several recent papers (see for instance [42] and [43]). This regime, corresponding to large norm initializations, is also characterized by low accuracy in terms of the expected error. Can we explain this behavior in terms of our formalization of the dynamics? A more general question is of key interest also for applications: can we characterize the conditions that ensure convergence to the "best" of the maximum margin solutions? Less important but still interesting is the question of why batch normalization is empirically better than weight normalization? This requires some explanation because both enforce implicitly or explicitly a unit constraint in the L_2 norm.

Acknowledgments

We thank Yuan Yao, Misha Belkin, Jason Lee and especially Sasha Rakhlin for illuminating discussions. Part of the funding is from Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216, and part by C-BRIC, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

References

[1] P. Niyogi and F. Girosi. On the relationship between generalization error, hypothesis complexity, and sample complexity for radial basis functions. *Neural Computation*, 8:819–842, 1996.

[2] T. Poggio and S. Smale. The mathematics of learning: Dealing with data. *Notices of the American Mathematical Society (AMS)*, 50(5):537–544, 2003.

[3] Moritz Hardt and Tengyu Ma. Identity matters in deep learning. *CoRR*, abs/1611.04231, 2016.

[4] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning. *arXiv:1706.08947*, 2017.

[5] Jure Sokolic, Raja Giryes, Guillermo Sapiro, and Miguel Rodrigues. Robust large margin deep neural networks. *arXiv:1605.08254*, 2017.

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

[7] C. Zhang, Q. Liao, A. Rakhlin, K. Sridharan, B. Miranda, N.Golowich, and T. Poggio. Musings on deep learning: Optimization properties of SGD. *CBMM Memo No. 067*, 2017.

[8] D. Soudry, E. Hoffer, and N. Srebro. The Implicit Bias of Gradient Descent on Separable Data. *ArXiv e-prints*, October 2017.

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

[10] F.M. Arscott and A.F. Filippov. *Differential Equations with Discontinuous Righthand Sides: Control Systems*. Mathematics and its Applications. Springer Netherlands, 1988.

[11] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M. Kakade, and Michael I. Jordan. How to escape saddle points efficiently. *CoRR*, abs/1703.00887, 2017.

[12] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points - online stochastic gradient for tensor decomposition. *CoRR*, abs/1503.02101, 2015.

[13] Jason D. Lee, Max Simchowitz, Michael I. Jordan, and Benjamin Recht. Gradient descent only converges to minimizers. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 1246–1257, Columbia University, New York, New York, USA, 23–26 Jun 2016. PMLR.

[14] Simon S. Du, Jason D. Lee, and Yuandong Tian. When is a convolutional filter easy to learn? In *International Conference on Learning Representations*, 2018.

[15] Yuandong Tian. An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 3404–3413. JMLR.org, 2017.

[16] M. Soltanolkotabi, A. Javanmard, and J. D. Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 65(2):742–769, Feb 2019.

[17] Yuanzhi Li and Yang Yuan. Convergence analysis of two-layer neural networks with relu activation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 597–607, USA, 2017. Curran Associates Inc.

[18] Alon Brutzkus and Amir Globerson. Globally optimal gradient descent for a convnet with gaussian inputs. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 605–614, 2017.

[19] Simon Du, Jason Lee, Yuandong Tian, Aarti Singh, and Barnabas Poczos. Gradient descent learns one-hidden-layer CNN: Don't be afraid of spurious local minima. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1339–1348, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[20] Simon S. Du, Jason D. Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. *CoRR*, abs/1811.03804, 2018.

[21] Kai Zhong, Zhao Song, Prateek Jain, Peter L. Bartlett, and Inderjit S. Dhillon. Recovery guarantees for one-hidden-layer neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 4140–4149. JMLR.org, 2017.

[22] Kai Zhong, Zhao Song, and Inderjit S. Dhillon. Learning non-overlapping convolutional neural networks with multiple kernels. *CoRR*, abs/1711.03440, 2017.

[23] X. Zhang, Y. Yu, L. Wang, and Q. Gu. Learning One-hidden-layer ReLU Networks via Gradient Descent. *arXiv e-prints*, June 2018.

[24] Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8157–8166. Curran Associates, Inc., 2018.

[25] 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.

[26] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic gradient descent optimizes over-parameterized deep relu networks. *CoRR*, abs/1811.08888, 2018.

[27] Amit Daniely. Sgd learns the conjugate kernel class of the network. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 2422–2430. Curran Associates, Inc., 2017.

[28] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *CoRR*, abs/1811.04918, 2018.

[29] Sanjeev Arora, Simon S. Du, Wei Hu, Zhi yuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *CoRR*, abs/1901.08584, 2019.

[30] Colin Wei, Jason D. Lee, Qiang Liu, and Tengyu Ma. On the margin theory of feedforward neural networks. *CoRR*, abs/1810.05369, 2018.

[31] 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.

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

[33] 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.

[34] Lorenzo Rosasco and Silvia Villa. Learning with incremental iterative regularization. In *Advances in Neural Information Processing Systems*, pages 1630–1638, 2015.

[35] Tim Salimans and Diederik P. Kingm. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *Advances in Neural Information Processing Systems*, 2016.

[36] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How Does Batch Normalization Help Optimization? *arXiv e-prints*, page arXiv:1805.11604, May 2018.

[37] Saharon Rosset, Ji Zhu, and Trevor Hastie. Margin maximizing loss functions. In *Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8-13, 2003, Vancouver and Whistler, British Columbia, Canada]*, pages 1237–1244, 2003.

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

[39] Alexander Rakhlin and Xiyu Zhai. Consistency of Interpolation with Laplace Kernels is a High-Dimensional Phenomenon. *arXiv e-prints*, page arXiv:1812.11167, Dec 2018.

[40] Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and double descent curve. *arXiv e-prints*, page arXiv:1908.05355, Aug 2019.

[41] Mikhail Belkin, Daniel Hsu, and Ji Xu. Two models of double descent for weak features. *CoRR*, abs/1903.07571, 2019.

[42] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A Mean Field View of the Landscape of Two-Layers Neural Networks. *arXiv e-prints*, page arXiv:1804.06561, Apr 2018.

[43] Phan-Minh Nguyen. Mean Field Limit of the Learning Dynamics of Multilayer Neural Networks. *arXiv e-prints*, page arXiv:1902.02880, Feb 2019.

[44] C. Zhang, Q. Liao, A. Rakhlin, K. Sridharan, B. Miranda, N.Golowich, and T. Poggio. Theory of deep learning IIb: Optimization properties of SGD. *CBMM Memo 072*, 2017.

[45] M. Raginsky, A. Rakhlin, and M. Telgarsky. Non-convex learning via stochastic gradient langevin dynamics: A nonasymptotic analysis. *arXiv:180.3251 [cs, math]*, 2017.

[46] T. Poggio and Q. Liao. Theory II: Landscape of the empirical risk in deep learning. *arXiv:1703.09833, CBMM Memo No. 066*, 2017.

[47] O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. pages 169–207, 2003.

[48] Peter L. Bartlett, Michael I. Jordan, and Jon D. McAuliffe. Convexity, classification, and risk bounds. Technical report, 2003.

[49] S. C. Douglas, S. Amari, and S. Y. Kung. On gradient adaptation with unit-norm constraints. *IEEE Transactions on Signal Processing*, 48(6):1843–1847, June 2000.

[50] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[51] Simon S Du, Wei Hu, and Jason D Lee. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 384–395. Curran Associates, Inc., 2018.

[52] Paulo Jorge S. G. Ferreira. The existence and uniqueness of the minimum norm solution to certain linear and nonlinear problems. *Signal Processing*, 55:137–139, 1996.

[53] Qianli Liao, Brando Miranda, Andrzej Banburski, Jack Hidary, and Tomaso A. Poggio. A surprising linear relationship predicts test performance in deep networks. *CoRR*, abs/1807.09659, 2018.

Appendix

6 The optimization landscape of (unnormalized) Deep RELU Networks under exponential-type loss

The *first part* of the argument of this section relies on the simple observation that RELU networks, under the hypothesis of an exponential-type loss function, do not have zeros of the gradient (wrt the W_k) that separate the data. In fact, under the hypothesis of an exponential-type loss, separable data and homogeneity of the network – such as kernel machines and deep RELU networks – the only stationary points of the gradient that separate the data are for $\rho = \infty$.

Notice that minima arbitrarily close to zero loss exist for any finite, large ρ. For $\rho \to \infty$, the Hessian becomes arbitrarily close to zero, with all eigenvalues being close to zero. On the other hand, any point of the loss at a finite ρ has a Hessian wrt W_k which is not identically zero: for instance in the linear case the Hessian is proportional to $\sum_n^N x_n x_n^T$.

Consider now that the local minima which are not global minima must missclassify. How degenerate are they? In the case of a linear network in the exponential loss case, assume there is a finite w for which the gradient is zero in some of its components. One question is whether this is similar to the regularization case or not, that is whether *misclassification regularizes*.

Let us look at a linear example:

$$\dot{w} = F(w) = -\nabla_w L(w) = \sum_{n=1}^{n} y_n x_n^T e^{-y_n x_n^T w} \tag{32}$$

in which we assume that there is one classification ' error (say for $n = 1$), meaning that the term $e^{-y_1 x_1^T w}$ grows exponentially with w. Let us also assume that gradient descent converges to w^*. This implies that $\sum_{n=2}^{n} y_n x_n^T e^{-y_n x_n^T w^*} = -y_1 x_1^T e^{-y_1 x_1^T w^*}$: for w^* the gradient is zero and $\dot{w} = 0$. Is this a convex equilibrium? Let us look at a very simple $1D$, $n = 2$ case:

$$\dot{w} = -x_1 e^{x_1 w^*} + x_2 e^{-x_2 w^*} \tag{33}$$

If $x_2 > x_1$ then $\dot{w} = 0$ for $e^{(x_1+x_2)w^*} = \frac{x_2}{x_1}$ which implies $w^* = \frac{\log(\frac{x_2}{x_1})}{x_1+x_2}$. This is clearly a hyperbolic equilibrium point, since we have

$$\nabla_w F(w) = -x_1^2 e^{x_1 w^*} - x_2^2 e^{-x_2 w^*} < 0, \tag{34}$$

so the single eigenvalue in this case has no zero real part.

In general, if there are only a small number of classification errors, one expects a similar situation for some of the components. *Differently from the regularization case, misclassification errors do not "regularize" all components of w but only the ones in the span of the misclassified examples.*

The more interesting case is with $D > N$. An example of this case is $D = 3$ and $N = 2$ in the above equation. The Hessian wrt W_k at the minimum will be degenerate with at least one zero eigenvalue ad one negative eigenvalue.

The stationary points of the gradient of f in the nonlinear multilayer separable case under exponential loss are given by

$$\sum_{n=1}^{N} y_n \frac{\partial f(W; x_n)}{\partial W_k^{i,j}} e^{-y_n f(W;x_n)} = 0. \tag{35}$$

This means that the global zeros of the loss are at infinity, that is for $\rho \to \infty$ in the exponential. If other stationary points were to exist for a value W^* of the weights, they would be given by zero-valued linear combinations with positive coefficients of $\frac{\partial f(W;x_n)}{\partial W_k^{i,j}}$. Use of the structural Lemma shows that $\frac{\partial f(W;x)}{\partial W_k^{i,j}} = 0, \forall i, j, k$ implies $f(W^*; x) = 0$. So stationary points of the gradient wrt W_k that are data-separating do not exist for any finite ρ. The situation is quite different if we consider *stationary points wrt* V_k.

Clearly, it would be interesting to characterize better the degeneracy of the local minima. For the goals of this section however the fact that they cannot be completely degenerate is sufficient. We thus have the following rather obvious result:

Theorem 4 *Under the exponential loss, the weight W_k for zero loss at infinite ρ are completely degenerate, with all eigenvalues of the Hessian being zero. The other stationary points of the gradient are less degenerate, with at least one nonzero eigenvalue.*

The *second part* of our argument (in [44]) is that SGD concentrates on the most degenerate minima. The argument is based on the fact that the Boltzman distribution is formally the asymptotic "solution" of the stochastic differential Langevin equation and also of SGDL, defined as SGD with added white noise (see for instance [45]. More informally, there is a certain similarity between SGD and SGDL suggesting that in practice the solution of SGD may be similar to the solution of SGDL. The Boltzman distribution is

$$p(W_k^{i,j}) = \frac{1}{Z} e^{-\frac{L(f)}{T}}, \tag{36}$$

where Z is a normalization constant, $L(f)$ is the loss and T reflects the noise power. The equation implies that SGDL prefers degenerate minima relative to non-degenerate ones of the same depth. In addition, among two minimum basins of equal depth, the one with a larger volume is much more likely in high dimensions as shown by the simulations in [44]. Taken together, these two facts suggest that SGD selects degenerate minimizers corresponding to larger isotropic flat regions of the loss. Then SDGL shows concentration – *because of the high dimensionality* – of its asymptotic distribution Equation 36.

Together [46] and [44] imply the following

Conjecture *For overparametrized deep networks under an exponential-type loss, SGD selects with high probability global minimizers of the empirical loss, which are fully degenerate (for $\rho \to \infty$, in the separable case.*

7 Uniform convergence bounds: minimizing a surrogate loss under norm constraint

Classical *generalization bounds for regression* [47] suggest that minimizing the empirical loss of a loss function such as the cross-entropy subject to constrained *complexity of the minimizer* is a way to to attain generalization, that is an expected loss close to the empirical loss:

Theorem 5 *The following generalization bounds apply to* $\forall f \in \mathbb{F}$ *with probability at least* $(1-\delta)$*:*

$$L(f) \leq \hat{L}(f) + c_1 \mathbb{R}_N(\mathbb{F}) + c_2 \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}} \tag{37}$$

where $L(f) = \mathbf{E}[\ell(f(x), y)]$ is the expected loss, $\hat{L}(f)$ is the empirical loss, $\mathbb{R}_N(\mathbb{F})$ is the empirical Rademacher average of the class of functions $\mathbb{F}$ measuring its complexity; c_1, c_2 are constants that depend on properties of the Lipschitz constant of the loss function, and on the architecture of the network.

Thus minimizing under a constraint on the Rademacher complexity a surrogate function such as the cross-entropy (which becomes the logistic loss in the binary classification case) will minimize an upper bound on the expected classification error because such surrogate functions are upper bounds on the $0-1$ function[4]. Calling $\rho \tilde{f} = f$, using the homogeneity of the network, one can use the following version of the bound above:

Theorem 6 $\forall f \in \mathbb{F}$ *with probability at least* $(1-\delta)$*:*

$$L(\rho \tilde{f}) \leq \hat{L}(\rho \tilde{f}) + \rho \mathbb{R}_N(\tilde{\mathbb{F}}) + c_2 \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}} \tag{38}$$

and to use ρ effectively as a parameter.

In this setup, $\tilde{f}$ is obtained by minimizing the exponential loss for $\rho \to \infty$ under a unit norm constraint on the weight matrices of $\tilde{f}$:

$$\lim_{\rho \to \infty} \arg \min_{||V_k||=1, \ \forall k} L(\rho \tilde{f}) \tag{39}$$

As it will become clear later, gradient descent techniques on the exponential loss automatically increase ρ to infinity.

In the following we explore the implications for deep networks of this classical approach to generalization.

[4] Furtermore the excess classification risk $R(f) - R^*$, where $R(f)$ is the classification error associated with f and R^* is the Bayes error [48], can be bounded by a monotonically increasing function of appropriate surrogate functions such as the exponential and the cross-entropy.

8 Constrained minimization of the exponential loss implies margin maximization

Though not critical for our approach to the question of generalization in deep networks it is interesting to observe that constrained minimization of the exponential loss implies margin maximization. This property relates our approach to the results of several recent papers [8, 31, 32]. Notice that our theorem 7 as in [37] is a *sufficient condition for margin maximization.* Sufficiency is not true for general loss functions. In fact [31] seems to require additional conditions for ensuring that the margin path converges to the optimization path.

To state the margin property more formally, we adapt to our setting a different result due to [37] (they consider a vanishing λ regularization term whereas we have a unit norm constraint). First we recall the definition of the empirical loss $L(f) = \sum_{n=1}^{N} \ell(y_n f(x_n))$ with an exponential loss function $\ell(yf) = e^{-yf}$. We define $\eta(f)$ a the *margin* of f, that is $\eta(f) = \min_n f(x_n)$.

Then our margin maximization theorem takes the form

Theorem 7 *Consider the set of $V_k, k = 1, \cdots, K$ corresponding to*

$$\min_{||V_k||=1} L(f(\rho_k, V_k)) \tag{40}$$

where the norm $||V_k||$ is a chosen L_p norm and $L(f)(\rho_k, V_K) = \sum_n \ell(y_n \rho f(V; x_n))$ is the empirical exponential loss. For each layer k consider a sequence of increasing ρ. Then the associated sequences of V_k defined by Equation 40, converges for $\rho_k \to \infty$ to the maximum margin of f, that is to $\max_{||V_k||\leq 1} \eta(f)$.

This means that for $\rho \to \infty$ the weights V_k that minimize the exponential loss under a unit norm constraint converge to the weights V_k that maximize the margin of $f(V; x)$. Thus

$$\arg\min_{||V_k||=1,\rho} L(\rho f)(V; x_n) \to \arg \max_{||V_k||=1} \min_n y_n f(V; x_n) \tag{41}$$

Proof The proof loosely follows [37, 8], see also [30]. We observe that $\min_{||V_k||=1} L(f(\rho))$ exists because $L(f(\rho))$ is continuous since f is continuous and the domain is compact for any finite ρ_k. We carry on the argument for a specific k with the understanding that the same steps should be done for each k. In the following we drop k for simplicity of notation. We now assume that V^* minimize $L(f(\rho))$. We claim that the associated network f^* maximizes the margin for $\rho \to \infty$. Arguing by contradiction assume that for a given ρ there exist a different f_1 with a larger margin, that is $\eta(f_1) > \eta(f^*)$. Then we can choose any f_2 with weights V_2 such that $||V_2 - V^*|| \leq \delta$ such that $\eta(f_2) < \eta(f_1) - \epsilon$. Now we can choose a ρ large enough so that ρf_1 has a smaller loss then ρf_2, implying that f^* cannot be a convergence point. The last step is based on the fact that *if $\eta(f_1) > \eta(f_2)$, then $L(\rho f_1) < L(\rho f_2)$ for large enough ρ.*

This follows from the existence of ρ such that $L(\rho y_1 f_1) \leq N e^{-\rho \eta(f_1)} \leq e^{-\rho \eta(f_2)} \leq L(\rho y_2 f_2)$.

Theorem 7 can be supplemented with the following lemma, proved in the following Appendix 9:

Lemma 8 *Margin maximization of the network with normalized weights is equivalent to norm minimization under strict separability ($y_n f(x_n) \geq 1$).*

9 Minimal norm and maximum margin

We discuss the connection between maximum margin and minimal norms problems in binary classification. To do so, we reprise some classic reasonings used in the context of support vector machines. The norm assumed in this section is the L_2 norm (the proof can be extended to other norms). We assume functions with the one-homogeneity property, namely, for all $\alpha > 0$,

$$f(\alpha W; x) = \alpha f(W; x)$$

. We also assume separability, that is $y_n f(x_n) > 0, \quad \forall n$.

Given a training set of N data points $(x_i, y_i)_{i=1}^N$, where labels are ± 1, the functional margin is defined as

$$\min_{i=1,\ldots,N} y_i f(W_K \cdots W_1; x_i). \tag{42}$$

The maximum (max) margin problem is then

$$\max_{W_K,\cdots,W_1} \min_i y_i f(W_K, \cdots, W_1; x_i), \qquad \text{subj. to} \quad \|W_k\| = 1, \quad \forall k. \tag{43}$$

The latter constraint is needed to avoid trivial solutions in light of the one-homogeneity property. We next show that

Theorem 9 *Problem* (43) *is equivalent to*

$$\min_{W_k} \frac{1}{2}\|W_k\|^2, \qquad \forall k \quad \textit{subj. to} \quad y_i f(W_K, \cdots, W_1; x_i) \geq 1, \qquad i = 1, \ldots, N. \tag{44}$$

To see this, we introduce a number of equivalent formulations. First, notice that functional margin (42) can be equivalently written as

$$\max_{\gamma > 0} \gamma, \qquad \text{subj. to} \quad y_i f(W_K, \cdots, W_1; x_i) \geq \gamma, \qquad i = 1, \ldots, N.$$

Then, the max margin problem (43) can be written as

$$\max_{W_K,\cdots,W_1,\gamma>0} \gamma, \qquad \text{subj. to} \quad \|W_k\| = 1, \quad \forall k \quad y_i f(W_K, \cdots, W_1; x_i) \geq \gamma, \qquad i = 1, \ldots, N. \tag{45}$$

Next, we can incorporate the norm constraint noting that using one-homogeneity,

$$y_i f(W_K, \cdots, W_1; x_i) \geq \gamma \Leftrightarrow y_i f(\frac{W_K}{\|W_K\|}, \cdots, \frac{W_1}{\|W_1\|}; x_i) \geq \gamma\rho \Leftrightarrow y_i f(W_K, \cdots, W_1; x_i) \geq \gamma'$$

so that Problem (45) becomes

$$\max_{W_K,\cdots,W_1;\gamma'>0} \frac{\gamma'}{\rho}, \qquad \text{subj. to} \qquad y_i f(W_K,\cdots,W_1;x_i) \geq \gamma', \qquad i=1,\ldots,N. \tag{46}$$

Finally, using again one-homogeneity, without loss of generality, we can set $\gamma'=1$ and obtain the equivalent problem

$$\max_{W_K,\cdots,W_1} \frac{1}{\rho}, \qquad \text{subj. to} \qquad y_i f(W_K,\cdots,W_1;x_i) \geq 1, \qquad i=1,\ldots,N. \tag{47}$$

The result is then clear noting that

$$\max_{W} \frac{1}{\rho} \Leftrightarrow \min_{||W_K||,\cdots,||W_1||} \rho \Leftrightarrow \min_{\rho_1,\cdots,\rho_K} \frac{\rho^2}{2}.$$

10 Gradient techniques for norm control

There are several ways to implement the minimization in the tangent space of $||V||^2=1$. In fact, a review of gradient-based algorithms with unit-norm constraints [49] lists

1. the *Lagrange multiplier method*
2. the *coefficient normalization method*
3. the *tangent gradient method*
4. the *true gradient method* using natural gradient.

For small values of the step size, the first three techniques are equivalent to each other and are also good approximations of the true gradient method [49]. The four techniques are closely related and have the same goal: performing gradient descent optimization with a unit norm constraint.

Remarks

- Stability issues for numerical implementations are discussed in [49].
- Interestingly, there is a close relationship between the Fisher-Rao norm and the natural gradient [9]. In particular, the natural gradient descent is the steepest descent direction induced by the Fisher-Rao geometry.
- Constraints in optimization such as $||v||_p=1$ imposes a geometric structure to the parameter space. If $p=2$ the weight vectors satisfying the unit norm constraint form a n-dimensional hypersphere of radius = 1. If $p=\infty$ they form an hypercube. If $p=1$ they form a hyperpolyhedron.

10.1 Lagrange multiplier method

We start with one of the techniques, the Lagrange multiplier method, because it enforces the unit constraint in an especially transparent way. We assume separability and incorporate the constraint in the exponential loss by defining a new loss as

$$L = \sum_{n=1}^{N} e^{-\rho f(V;x_n)y_n} + \sum_{k=1}^{K} \lambda_k(||V_k||_p^p - 1) \tag{48}$$

where the Lagrange multipliers λ_k are chosen to satisfy $||V_k||_p = 1$ at convergence or when the algorithm is stopped.

We perform gradient descent on L with respect to ρ, V_k. We obtain for $k = 1, \cdots, K$

$$\dot{\rho_k} = \sum_n \frac{\rho}{\rho_k} e^{-\rho(t) f(V;x_n)y_n} y_n f(V;x_n), \tag{49}$$

and for each layer k

$$\dot{V_k} = \rho(t) \sum_n e^{-\rho(t) y_n f(V;x_n)} y_n \frac{\partial f(V;x_n)}{\partial V_k}(t) + \lambda_k(t) p V_k^{p-1}(t). \tag{50}$$

The sequences $\lambda_k(t)$ must satisfy $\lim_{t\to\infty} ||V_k||_p = 1 \quad \forall k$.
Remarks

1. In the case of $p = 2$, with the conditions $||V_k|| = 1$ at each t, $\lambda_k(t)$ must satisfy

$$||V_k(t) + \rho(t) \sum_n e^{-\rho(t) y_n f(V;x_n)} y_n \frac{\partial f(V;x_n)}{\partial V_k} - 2\lambda_k(t) V_k(t)|| = 1. \tag{51}$$

Thus defining $g(t) = \rho(t) \sum_n e^{-\rho(t) y_n f(V;x_n)} y_n \frac{\partial f(V;x_n)}{\partial V_k}(t)$ we obtain

$$||V_k(t) + g(t) + 2\lambda_k(t) V_k(t)|| = 1, \tag{52}$$

that is

$$||\alpha(t) V_k(t) + g(t)|| = 1, \tag{53}$$

with $\alpha(t) = 1 + 2\lambda_k(t)$. The solution for α is

$$\alpha(t) = \sqrt{1 - ||g(t)||^2 + (V_k^T g(t))^2} - V^T(t) g(t). \tag{54}$$

Thus λ goes to zero at infinity because $g(t)$ does and $\alpha \to 1$.

2. Since the first term in the right hand side of Equation (50) goes to zero with $t \to \infty$ and the Lagrange multipliers λ_k also go to zero, the normalized weight vectors converge at infinity to $\dot{V}_k = 0$. On the other hand, $\rho(t)$ grows to infinity. As shown in section 4.5, the norm square ρ_k^2 (when $p = 2$) of each layer grows at the same rate.

3. As in the case of the of the vanishing regularizer assumed in the original theorem of [37], the Lagrange multipliers λ_k here go to zero.

4. The Lagrange multiplier approach with $\lambda_k \to 0$ establishes a connection with Halpern iterations and minimum norm solutions for degenerate minima.

10.2 Coefficient normalization method

If $u(k)$ is unconstrained the gradient maximization of $L(u)$ with respect to u can be performed using the algorithm

$$u(k+1) = u(k) + g(k) \tag{55}$$

where $g(k) = \mu(k)\nabla_u L$. Such an update, however, does not generally guarantee that $||u^T(k+1)|| = 1$. The coefficient normalization method employs a two step update $\hat{u}(k+1) = u(k) + g(k)$ and $u(k+1) = \frac{\hat{u}(k+1)}{||\hat{u}(k+1)||_p}$.

10.3 Tangent gradient method

Theorem 10 *([49]) Let $||u||_p$ denote a vector norm that is differentiable with respect to the elements of u and let $g(t)$ be any vector function with finite L_2 norm. Then, calling $v(t) = \frac{\partial ||u||_p}{\partial u}_{u=u(t)}$, the equation*

$$\dot{u} = h_g(t) = Sg(t) = (I - \frac{vv^T}{|v||_2^2})g(t) \tag{56}$$

with $||u(0)|| = 1$, describes the flow of a vector u that satisfies $||u(t)||_p = 1$ for all $t \geq 0$.

In particular, a form for g is $g(t) = \mu(t)\nabla_u L$, the gradient update in a gradient descent algorithm. We call $Sg(t)$ the tangent gradient transformation of g. For more details see [49].

In the case of $p = 2$ we replace v in Equation 56 with u because $v(t) = \frac{\partial ||u||_2}{\partial u} = u$. This gives $S = (I - \frac{uu^T}{|u||_2^2})$ and $\dot{u} = Sg(t)$.

Remarks

- For $p = 2$ $v = \frac{\partial ||u||_p}{\partial u}_u$ is $v = \frac{u}{||u||_2}$
- For $p = 1$, $\frac{\partial ||u||_1}{\partial u_j} = \frac{u_j}{|u_j|}$.
- For $p = \infty$, $\frac{\partial ||u||_\infty}{\partial u_j} = \text{sign}(u_k)\delta_{k,j}$, if maximum is attained in coordinate k.

11 Standard dynamics, Weight Normalization and Batch Normalization

We now discuss the relation of some existing techniques for training deep networks with the gradient descent techniques under unit norm constraint of the previous section.

11.1 Standard unconstrained dynamics

The standard gradient dynamics is given by

$$\dot{W}_k^{i,j} = -\frac{\partial L}{\partial W_k^{i,j}} = \sum_{n=1}^{N} y_n \frac{\partial f(W; x_n)}{\partial W_k^{i,j}} e^{-y_n f(W; x_n)} \tag{57}$$

where W_k is the weight matrix of layer k. As we observed, this dynamics has global minima at infinity with zero loss, if the data are separable. The other stationary points have loss greater than zero.

Empirical observations suggest that unconstrained gradient dynamics on deep networks converges to solutions that generalize, especially when SGD is used instead of GD. In our experiments, normalization at each iteration, corresponding to the coefficient normalization method, improves generalization but not as much as Weight Normalization does.

11.2 Weight Normalization

For each layer (for simplicity of notation and consistency with the original weight normalization paper), weight normalization [35] defines v and g in terms of $w = g\frac{v}{|v|}$. The dynamics on g and v is induced by the gradient dynamics of w as follows (assuming $\dot{w} = -\frac{\partial L}{\partial w}$

$$\dot{g} = \frac{v^T}{||v||}\dot{w} \tag{58}$$

and

$$\dot{v} = \frac{g}{||v||} S\dot{w} \tag{59}$$

with $S = I - \frac{vv^T}{||v||^2}$.

We claim that this is the same dynamics obtained from tangent gradient for $p = 2$. In fact, compute the flows in ρ, v from $w = \rho v$ as

$$\dot{\rho} = \frac{\partial w}{\partial \rho}\frac{\partial L}{\partial w} = v^T\dot{w} \tag{60}$$

and

$$\dot{v} = S\rho\dot{w} \tag{61}$$

Clearly the dynamics of this algorithm is the same as standard weight normalization if $||v||_2 = 1$, because then Equations 58 and 59 become identical to Equations 60 and 61 with g corresponding to ρ. We now observe, multiplying Equation 59 by v^T, that $v^T\dot{v} = 0$ because $v^T S = 0$, implying that $||v||^2$ is constant in time. Thus if $||v|| = 1$ at initialization, it will not change (at least in the noiseless case). Thus *the dynamics of Equations 58 and 59 is the same dynamics as Equations 60 and 61.* It is also easy to see that the dynamics above is not equivalent to the standard dynamics on the w (see also Figure 3).

11.3 Batch Normalization

Batch normalization [50] for unit i in the network normalizes the input vector of activities to unit i – that is it normalizes $X^j = \sum_j W^{i,j} x_j$, where x_j are the activities of the previous layer. Then it sets the activity to be

$$Y^j = \gamma \cdot \hat{X}^j + \beta = \gamma \frac{X^j - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta,$$

where γ, β are learned subsequently in the optimization and

$$\mu_B = \frac{1}{N} \sum_{n=1}^{N} X_n \qquad \sigma_B^2 = \frac{1}{N} \sum_{n=1}^{N} (X_n - \mu_B)^2.$$

Note that both μ_B and σ_B^2 are vectors, so the division by $\sqrt{\sigma_B^2 + \epsilon}$ has to be understood as a point-wise Hadamard product $\odot(\sigma_B^2 + \epsilon)^{-1/2}$. The gradient is taken wrt the new activations defined by the transformation above.

Unlike Weight Normalization, the Batch Normalization equations do not include an explicit computation of the partial derivatives of L with respect to the new variables in terms of the standard gradient $\frac{\partial L}{\partial w}$. The reason is that Batch Normalization works on an augmented network: a BN module is added to the network and partial derivatives of L with respect to the new variables are directly computed on its output. Thus the BN algorithm uses only the derivative of L wrt the old variables as a function of the derivatives of L wrt new variables in order to update the parameters below the BN module by applying the chain rule. Thus we have to estimate what BN *implies* about the partial derivatives of L with the respect to the new variables as a function of the standard gradient $\frac{\partial L}{\partial w}$.

To see the nature of the dynamics implied by batch normalization we simplify the original Equations (in the Algorithm 1 box in [50]). Neglecting μ_B and β and γ, we consider the core transformation as $\hat{X} = \frac{X}{\sigma_B}$ which, assuming fixed inputs, becomes $\hat{X} = \frac{X}{|X|}$ which is mathematically identical with the transfomation of section 11.1 $v = \frac{w}{|w|}$. In a similar way the dynamics of $w = \frac{\partial L}{\partial w}$ induces the following dynamics on $\hat{X}$:

$$\dot{\hat{X}} = \frac{\partial \hat{X}}{\partial X} \dot{X} \tag{62}$$

where $\dot{x} = \nabla_x L$. We consider $X \in \mathbb{R}^{N \times D}$. In the $D = 1$ case, we get

$$\frac{\partial \hat{X}}{\partial X} = (\sigma_B^2 + \epsilon)^{-1/2} \left[-\frac{1}{N} \hat{X} \hat{X}^T + I \right].$$

In the general D-dimensional vector case, this generalizes to

$$\frac{\partial \hat{X}}{\partial X} = (\sigma_B^2 + \epsilon)^{-1/2} \left[-\frac{1}{N} \hat{X}^T \odot \hat{X} + I \right].$$

Notice that $I - \hat{X}\hat{X}^T = S$. Since $x = Wx_{input}$ this shows that batch normalization is closely related to *gradient descent algorithms with unit L_2 norm constraint of the tangent gradient type*. Because of the simplifications we made, there are other differences between BN and weight normalization, some of which are described in the remarks below.

Remarks

1. Batch normalization (see Supplementary Material), does not control directly the norms of $W_1, W_2, \cdots, W_K$ as WN does. Instead it controls the norms

$$||x||, ||\sigma(W_1 x)||, ||\sigma(W_2\sigma(W_1 x))||, \cdots \tag{63}$$

   In this sense it implements a somewhat weaker version of the generalization bound.

2. In the multilayer case, BN controls separately the norms $||V_i||$ of the weights into unit i, instead of controlling the overall Frobenius norm of the matrix of weights as WN does. Of course control of the $||V_i||$ implies control of $||V||$ since $||V||^2 = \sum_i ||V_i||_i^2$.

11.4 Weight Normalization and Batch Normalization enforce an explicit unit 2-norm constraint

Consider the *tangent gradient transformation* to a gradient increment $g(t) = \mu(k)\nabla_u L$ defined as $h_g = Sg(t)$ with $S = I - \frac{uu^T}{||u||_2^2}$. Theorem 10 says that the dynamical system $\dot{u} = h_g$ with $||u(0)||_2 = 1$ describes the flow of a vector u that satisfies $||u(t)|_2 = 1$ for all $t \geq 0$. It is obvious then that

Observation 1 *The dynamical system describing weight normalization (Equations 58 and 59) are not changed by the tangent gradient transformation.*

The proof follows easily for WN by using the fact that $S^2 = S$. The same argument can be applied to BN. The property is consistent with the statement that they enforce an L_2 unit norm constraint.

Thus all these techniques implement margin maximization of f_V under unit norm constraint of the weight matrices of f_V. Consider for instance the Lagrange multiplier method. Let us assume that starting at some time t, $\rho(t)$ is large enough that the following asymptotic expansion (as $\rho \to \infty$) is a good approximation: $\sum_n e^{-\rho(t)y_n f(V;x_n)} \sim C \max_n e^{-\rho(t)y_n f(V;x_n)}$, where C is the multiplicity of the x_n with minimal value of the margin of $f(V; x_n)$. The data points with the corresponding minimum value of the margin $y_n f(V; x_n)$ are called support vectors (the x_i, y_i s.t $\arg\min_n y_n f(V; x_n)$). They are a subset of cardinality C of the N datapoints, all with the same margin η. In particular, the term $g(t) = \rho(t) \sum_n e^{-\rho(t)y_n f(V;x_n)} y_n \frac{\partial f(V;x_n)}{\partial V_k}$ becomes $g(t) \approx \rho(t) e^{-\rho(t)\eta} \sum_i^C y_i \frac{\partial f(V;x_i)}{\partial V_k}$.

As we mentioned, in GD with unit norm constraint there will be convergence to $\dot{V}_k = 0$ for $t \to \infty$. There may be trajectory-dependent, multiple alternative selections of the support vectors (SVs) during the course of the iteration while ρ grows: each set of SVs may correspond

to a max margin, minimum norm solution without being the global minimum norm solution. Because of Bezout-type arguments [46] *we expect multiple maxima*. They should generically be degenerate even under the normalization constraints – which enforce each of the K sets of V_k weights to be on a unit hypersphere. Importantly, the normalization algorithms ensure control of the norm and thus of the generalization bound even if they cannot ensure that the algorithm converges to the globally best minimum norm solution (this depends on initial conditions for instance).

12 Dynamics of ρ

First we show that for each layer $\frac{\partial \rho_k^2}{\partial t}$ is the same irrespectively of k. Then we consider the dynamics of ρ. In the 1-layer network case the ρ dynamics yields $\rho \approx \log t$ asymptotically. For deeper networks, we will show that the product of weights at each layer diverges faster than logarithmically, but each individual layer diverges slower than in the 1-layer case.

12.1 The rate of growth of ρ_k is the same for all layers

A property of the dynamics of W_k, shared with the dynamics of V_k under explicit unit norm constraint, is suggested by recent work [51]: *the rate of change of the squares of the Frobenius norms of the weights of different layers is the same during gradient descent.* This implies that if the weight matrices are small at initialization, the gradient flow corresponding to gradient descent maintains approximatevely equal Frobenius norms across different layers, which is *a consequence of the norm constraint.* This property is expected in a minimum norm situation, which is itself equivalent to maximum margin under unit norm (see Appendix 9). The observation of [51] is easy to prove in our framework. Consider the gradient descent equations

$$\dot{W}_k^{i,j} = \sum_{n=1}^{N} y_n [\frac{\partial f(W; x_n)}{\partial W_k^{i,j}}] e^{-y_n f(W; x_n)}. \tag{64}$$

We use the relation $||\dot{W}_k|| = \frac{\partial ||W_k||}{\partial W_k} \frac{\partial W_k}{\partial t} = \frac{W_k}{||W_k||} \dot{W}_k$ by multiplying both sides of Equation 64 by W_k, summing over i, j and using lemma 3. Thus we obtain the following dynamics on $||W_k||$:

$$||\dot{W}_k|| = \frac{1}{||W_k||} \sum_{n=1}^{N} y_n f(W; x_n) e^{-y_n f(W; x_n)}. \tag{65}$$

It follows that

$$||\dot{W}_k||^2 = 2 \sum_{n=1}^{N} y_n f(W; x_n) e^{-y_n f(W; x_n)}, \tag{66}$$

which shows that the rate of growth of $||W_k||^2$ is independent of k. If we assume that $||W_1|| = ||W_2|| = \cdots = ||W_K|| = \rho_1(t)$ initially, they will remain equal while growing throughout training. The norms of the layers are balanced, thus avoiding the situation in which one layer may

contribute to decreasing loss by improving $\tilde{f}$ but another may achieve the same result by simply increasing its norm.

12.2 Rate of growth of weights

In linear 1-layer networks the dynamics of gradient descent yield $\rho \sim \log t$ asymptotically. For the validity of the results in the previous section, we need to show that the weights of a deep network also diverge at infinity. In general, the K nonlinearly coupled equations are not easily solved analytically. For simplicity of analysis, let us consider the case of a single training example $N = 1$, as we expect the leading asymptotic behavior to be independent of N. In this regime we have

$$\rho_k \dot{\rho}_k = yf(V;x) \left(\prod_{i=1}^{k} \rho_i \right) e^{-\prod_{i=1}^{K} \rho_i yf(V;x)} \tag{67}$$

Keeping all the layers independent makes it difficult to disentangle for example the behavior of the product of weights $\prod_{i=1}^{K} \rho_i$, as even in the 2-layer case the best we can do is to change variables to $r^2 = \rho_1^2 + \rho_2^2$ and $\gamma = e^{\rho_1 \rho^2 f_V(x)}$, for which we still get the coupled system

$$\dot{\gamma} = yf(V;x)^2 r^2, \qquad r\dot{r} = 2\frac{\log \gamma}{\gamma}, \tag{68}$$

from which reading off the asymptotic behavior is nontrivial.

We consider the case $\rho := \rho_1 = \rho_2 = \ldots = \rho_k$. This turns out to be true in general (see Equation 66). It gives us the single differential equation

$$\dot{\rho} = yf(V;x) K \rho^{K-1} e^{-\rho_k yf(V;x)}. \tag{69}$$

This implies that for the exponentiated product of weights we have

$$\left(e^{\rho_k yf(V;x)} \right)^{\cdot} = f(V;x)^2 K^2 \rho^{2K-2}. \tag{70}$$

Changing the variable to $R = e^{\rho_k y f_V(x)}$, we get finally

$$\dot{R} = f_V(x)^{\frac{2}{K}} K^2 \left(\log R \right)^{2-\frac{2}{K}}. \tag{71}$$

We can now readily check that for $K = 1$ we get $R \sim t$, so $\rho \sim \log t$. It is also immediately clear that for $K > 1$ the product of weights diverges faster than logarithmically. In the case of $K = 2$ we get $R(t) = \mathrm{li}^{-1}(yf(V;x)K^2 t + C)$, where $\mathrm{li}(z) = \int_0^z dt/\log t$ is the logarithmic integral function. We show a comparison of the 1-layer and 2-layer behavior in the left graph in Figure 4. For larger K we get faster divergence, with the limit $K \to \infty$ given by $R(t) = \mathcal{L}^{-1}(\alpha_\infty t + C)$, where $\alpha_\infty = \lim_{K\to\infty} (yf(V;x))^{\frac{2}{K}} K^2$ and $\mathcal{L}(z) = \mathrm{li}(z) - \frac{z}{\log z}$.

Interestingly, while the product of weights scales faster than logarithmically, the weights at each layer diverge slower than in the linear network case, as can be seen in the right graph in Figure 4.

13 Critical points of the flow

We discuss here in more details the stationary points of the gradient flow.

The critical points of the flow induced by the Lagrange multiplier method described in section 10.1, $\dot{V}_k(t) = 0$ are given by the following set of equations – *as many as the number of weights*:

$$\dot{V}_k = \rho \sum_n e^{-\rho y_n f(V;x_n)} y_n \frac{\partial f(V;x_n)}{\partial V_k} - 2\lambda_k(\rho) V_k. \tag{72}$$

The critical points of the unconstrained gradient system $\dot{V}_k(t) = 0$ can be rewritten as (dropping here the subscript k in V_k for simplicity of notation)

$$0 = \sum_n e^{-y_n f(x_n)} (I - VV^T) \frac{\partial f(V;x_n)}{\partial V}. \tag{73}$$

As it turns out, the evaluation of the Hessian of a linear un-normalized network is simple and shows that the global minima are hyperbolic and unique [8][5]

For nonlinear, multilayer networks there are multiple minima for finite ρ and it seems difficult to state that critical points of the flow – after separability is reached – are or are not hyperbolic minima.

We argue however that some of them must be convex minima. The argument is as follows:

- After separability and after a single support vector (or group of SVs with same margin) dominates, the margin increases until a critical point of the flow is reached (see lemma 11 below).
- If the rather weak conditions of [13] are valid, then we can argue that gradient descent (and especially SGD) do not usually get stuck in saddle points. and that therefore an asymptotic critical point of the flow is likely to correspond to a convex minimum.

Lemma 11 *If, after separation is reached, there is a sufficiently large ρ for which the sum $\sum_{n=1}^N e^{-\rho y_n f(V;x_n)} \propto e^{-\rho y_* f(V;x_*)}$ then the margin increases $y_* \frac{\partial f(V;x_*)}{\partial t} \geq 0$ (even if ρ is kept fixed).*

Proof $y_* f(V; x*)$ increases monotonically or is constant because

$$y_* \frac{\partial f(V;x*)}{\partial t} = \sum_k (\frac{\partial y_* f(V;x*)}{\partial V_k})^T \dot{V}_k = \sum_k \frac{\rho}{\rho_k^2} e^{-\rho y_* f(V;x^*)} \left(\left\| \frac{\partial f(V;x^*)}{\partial V_k} \right\|_F^2 - f(V;x^*)^2 \right). \tag{75}$$

[5] An easy calculation shows that Hessian wrt weights w is

$$H_{lin} = \sum_n^N x_n x_n^T e^{-x_n^T w}, \tag{74}$$

which is a sum of outer products. If the data span the space, this leads to a non-degenerate Hessian.

Equation 75 implies $\frac{\partial y_* f(V,x_*)}{\partial t} \geq 0$ because $||f(V;x^*)||_F = ||V_k^T \frac{\partial f(V;x^*)}{\partial V_k}||_F \leq ||\frac{\partial f(V;x^*)}{\partial V_k}||_F$, since the Frobenius norm is sub-multiplicative and V_k has unit norm. We used Equation 26 that is $\dot{V}_k = \frac{\rho}{\rho_k^2} \sum_{n=1}^N e^{-\rho y_n f(V;x_n)} y_n (\frac{\partial f(V;x_n)}{\partial V_k} - V_k f(V;x_n))$.

Notice that we do not have a proof that the stationary point can be realized – in other words we do not know if f can satisfy the equations $0 = (I - V_k V_k^T) \frac{\partial f(V;x^*)}{\partial V_k}$ for all k. The equations are consistent however for ReLU networks as one can check by multiplying on the left by V^T.

13.0.1 The Hessian has at least one positive eigenvalue

We discuss here in more details the stationary points of the gradient flow.

The critical points of the flow $\dot{V}_k(t) = 0$ are given by the following set of equations – *as many as the number of weights*:

$$\dot{V}_k = \rho \sum_n e^{-\rho y_n f(V;x_n)} y_n \frac{\partial f(V;x_n)}{\partial V_k} - 2\lambda_k(\rho) V_k. \tag{76}$$

The critical points of the unconstrained gradient system $\dot{V}_k(t) = 0$ can be rewritten as (dropping here the subscript k in V_k for simplicity of notation)

$$\sum_n e^{-y_n f(x_n)} y_n (I - VV^T) \frac{\partial f(V;x_n)}{\partial V}. \tag{77}$$

The equations give the same zeros since they only differ by a positive factor ρ – that we assume fixed here.

We consider the Hessian of L associated with the dynamical system $\dot{V}_k = -F(V_k, \rho_k) = -\nabla_{V_k} L$ w.r.t. V_k. The Hessian tells us about the linearized dynamics around the asymptotic critical point of the gradient and thus about the sufficient conditions for local minima under the norm constraint. The question is whether $-H$ is negative semidefinite or negative definite. The second case is needed to have *sufficient conditions* for local minima. We consider the quadratic form $\sum_{k,k'}^{K,K} V_k^T H V_{k'}$, where V_k correspond to critical points of the gradient (the Hessian for the unconstrained and the constrained system differ by a factor ρ). This represent one specific direction only. We will show that the Heassian has positive eigenvalues for v

The negative Hessian for the standard gradient descent case is

$$-H_{k,k'}^{i,j} = \sum_n e^{-y_n \rho f(V;x_n)} (A_n + B_n + C_n) \tag{78}$$

with

$A_n = (-\rho \frac{\partial f(V;x_n)}{\partial V_k^i} (I - V_{k'}^j (V_{k'}^T)^\ell) \frac{\partial f(V;x_n)}{\partial V_{k'}^\ell})$

and

$B_n = y_n \delta_{kk'} (-(\delta_{i,j} V_{k'}^\ell + \delta_{\ell,i} V_{k'}^i) \frac{\partial f(V;x_n)}{\partial V_{k'}^\ell})$

and

$C_n = (I - V_{k'}^j (V_{k'}^T)^\ell) \frac{\partial^2 f(V;x_n)}{\partial V_k^i \partial V_{k'}^\ell}$.

For the form $\sum_{k,k'}^{K,K} V_k^i (-H_{k,k'}^{i,j}) V_{k'}^j$ we find

- the first term disappears, because $(I - V^j_{k'}(V^T_{k'})^\ell)V^j_{k'} = 0$ (because $(I - V^j_{k'}(V^T_{k'})^\ell) = S$ and $SV = 0$).
- the second term gives $-2K\rho \sum_n e^{-\rho y_n f(V;x_n)} y_n f(V;x_n)$
- the third term disappears for the same reason as the first one.

Thus if gradient descent separates the data at T_0, for $t > T_0$ $-H$ is negative definite for any finite ρ in the v direction. It becomes negative semi-definite in the limit $\rho = \infty$. .

notice that convergence at finite time probably means a small number of support vectors with same value and so good generalization...add theorem!

13.1 New algorithms?

An examination of the gradient descent dynamics on exponential losses, discussed above, suggests experimenting with *a new family of algorithms* in which $\rho(t)$ is designed and controlled independently of the gradient descent on the V_k: we need ρ to grow in order to drive the exponential loss towards the ideal classification loss but our real interest is in maximizing the margin of $f(V;x)$. The ideal time course of $\rho(t)$ could be made adaptive to the data. It seems possible to test dynamics that may be quite different from $\rho \propto \log t$. Of course the role of ρ is complex: in the initial stages of GD it makes sense that the best solutions are solutions with $||V_k|| = 1$ and lowest exponential loss. Intuitively they correspond to solutions that separate and have minimum $||W_k||$.

14 Remark

In the linear one-layer case $f(V;x) = v^T x$, the stationary points of the gradient are given by $\sum \alpha_n \rho(t)(x_n - vv^T x_n)$. The Lagrange multiplier case is similar, giving $\sum \alpha_n \rho(t) x_n - \lambda v = 0$, with λ satisfying $\lambda^2 = \sum_n e^{2\rho y_n v^T x_n} \rho^2 y_n x_n$. Thus $\lambda \to 0$ for $\rho \to \infty$.

15 L_p norm constraint

To understand whether there exists an implicit complexity control in standard gradient descent of the weight directions, we check whether there exists an L_p norm for which unconstrained normalization is equivalent to constrained normalization.

From Theorem 10 we expect the constrained case to be given by the action of the following projector onto the tangent space:

$$S_p = I - \frac{\nu\nu^T}{||\nu||_2^2} \quad \text{with} \quad \nu_i = \frac{\partial ||w||_p}{\partial w_i} = \text{sign}(w_i) \circ \left(\frac{|w_i|}{||w||_p} \right)^{p-1} . \tag{79}$$

The constrained Gradient Descent is then

$$\dot{\rho_k} = V_k^T \dot{W}_k \quad \dot{V}_k = \rho_k S_p \dot{W}_k. \tag{80}$$

On the other hand, reparametrization of the unconstrained dynamics in the p-norm gives (following Equations 49 and 24)

$$\begin{aligned}\dot{\rho_k} &= \frac{\partial ||W_k||_p}{\partial W_k}\frac{\partial W_k}{\partial t} = \text{sign}(W_k) \circ \left(\frac{|W_k|}{||W_k||_p}\right)^{p-1} \cdot \dot{W}_k \\ \dot{V_k} &= \frac{\partial V_k}{\partial W_k}\frac{\partial W_k}{\partial t} = \frac{I - \text{sign}(W_k) \circ \left(\frac{|W_k|}{||W_k||_p}\right)^{p-1} W_k^T}{||W_k||_p^{p-1}}\dot{W}_k.\end{aligned} \tag{81}$$

These two dynamical systems are clearly different for generic p reflecting the presence or absence of a regularization-like constraint on the dynamics of V_k.

As we have seen however, for $p = 2$ the 1-layer dynamical system obtained by minimizing L in ρ_k and V_k with $W_k = \rho_k V_k$ under the constraint $||V_k||_2 = 1$, is the weight normalization dynamics

$$\dot{\rho_k} = V_k^T \dot{W}_k \quad \dot{V_k} = S\rho_k \dot{W}_k, \tag{82}$$

which is quite similar to the standard gradient equations

$$\dot{\rho_k} = V_k^T \dot{W}_k \quad \dot{v} = \frac{S}{\rho_k}\dot{W}_k. \tag{83}$$

16 Linear networks: rates of convergence

We consider here the linear case. We rederive some of the results by [8] in our gradient flow framework. In the separable case of a linear network ($f(x) = \rho v^T x$) the dynamics is

$$\dot{\rho} = \frac{1}{\rho}\sum_{n=1}^{N} e^{-\rho y_n v^T x_n} y_n v^T x_n \tag{84}$$

and

$$\dot{v} = \frac{1}{\rho}\sum_{n=1}^{N} e^{-\rho y_n v^T x_n} y_n (x_n - v v^T x_n). \tag{85}$$

As discussed earlier there are K support vectors with the same smallest margin. For t increasing, since $\rho \approx \log t$,

$$\dot{v} \propto \frac{1}{\rho}\sum_{j=1}^{K} \alpha_j (I - vv^T) x_j. \tag{86}$$

with $\alpha_j = y_j e^{-\rho y_j v^T x_j}$. If gradient descent converges, the solution v satisfies $vv^T x = x$, where $x = \sum_{j=1}^{K} \alpha_j^* x_j$. It is easy to see that $v = ||x||^2 x^\dagger$ – where $x^\dagger$ is the pseudoinverse of x – is a solution since the pseudoinverse of a non-zero vector z is $z^\dagger = \frac{z^T}{||z||^2}$. On the other hand, in the

linear case the operator T in $v(t+1) = Tv(t)$ associated with equation 86 is not expanding because v has unit norm. Thus [52] there is a fixed point $v = x$ which is *independent of initial conditions*.

To simplify the analysis we assume in the following that there is a single effective support vector (that is a set of points with the same margin). As one of the figures show, this is often not correct. The rates of convergence of the solutions $\rho(t)$ and $v(t)$, derived in different way in [8], may be read out from the equations for ρ and v. It is easy to check that a general solution for ρ is of the form $\rho \propto C \log t$. A similar estimate for the exponential term, gives $e^{-\rho y_n v^T x_n} \propto \frac{1}{t}$. We claim that a solution for the error $\epsilon = v - x$, since v converges to x, behaves as $\frac{1}{\log t}$. In fact we write $v = x + \epsilon$ and plug it in the equation for v in 86 (we also assume for notation convenience a single data point x). We obtain

$$\dot{\epsilon} = \frac{1}{\rho} e^{-\rho v^T x}(x - (x+\epsilon)(x+\epsilon)^T x) = \frac{1}{\rho} e^{-\rho v^T x}(x - x - x\epsilon^T - \epsilon x^T) \tag{87}$$

which has the form $\dot{\epsilon} = -\frac{1}{t \log t}(2x\epsilon^T)$. A solution of the form $\epsilon \propto \frac{1}{\log t}$ satisfies the equation: $-\frac{1}{t \log^2 t} = -B\frac{1}{t \log^2 t}$. Thus the error indeed converges as $\epsilon \propto \frac{1}{\log t}$.

A similar analysis for the weight normalization equations considers the same dynamical system with a change in the equation for v which becomes

$$\dot{v} \propto e^{-\rho} \rho (I - vv^T)x. \tag{88}$$

This equation differs by a factor ρ^2 from equation 87. As a consequence equation 88 is of the form $\dot{\epsilon} = -\frac{\log t}{t}\epsilon$, with a general solution of the form $\epsilon \propto t^{-1/2 \log t}$. Numerical experiments confirm these rates for linear networks with one data point, see Figure 5, but suggest modifications in the case with $N \neq 1$, see Figure 6. In summary, *GD with weight normalization converges faster to the same equilibrium than standard gradient descent: the rate for $\epsilon = v - x$ is $\frac{1}{t^{1/2 \log t}}$ vs $\frac{1}{\log t}$.*

Our simplified analysis implies that batch normalization has the same convergence rate as weight normalization (in the linear one layer case BN is identical to WN). As we discussed, it is however different wrt WN in the multilayer case: it has for instance separate normalization for each unit, that is for each row of the weight matrix at each layer.

17 Scaling ρ

With the observation that weight normalization leads to faster rates of convergence than unconstrained dynamics, it is natural to consider alternative scalings of ρ. In the dynamics of $\dot{V}$, We experiment with different rescalings $\alpha(t)$. The dynamics of normalized weights then become

$$\dot{V} = \alpha \frac{1}{\rho} \sum_{n=1}^{N} e^{-\alpha \rho y_n V^T x_n} y_n (x_n - VV^T x_n). \tag{89}$$

Specificially, we will consider $\alpha(t) = \frac{1}{\log(t)}$, $\log(t)$, $\log\log(t)$, $\exp(t)$. See Figure 8

Additionally, to better understand the effects of ρ on the test loss and error, we run experiments similar to [53] where networks were initialized with weights of different Gaussian standard deviations and a linear relationship between test loss and error emerges. Here, we scale ρ linearly to observe when this linear relationship disappears as ρ is increased.

18 Epilogue

The first observation is that $y = Wx$ can be made better conditioned by doing $y = ABx$ Suppose W is n, n; then A could be n, d and B d, n with $AB = W$ which means $A = WB^{\dagger}$ or $B = A^{\dagger}W$. Can I have better condition numbers with AB?

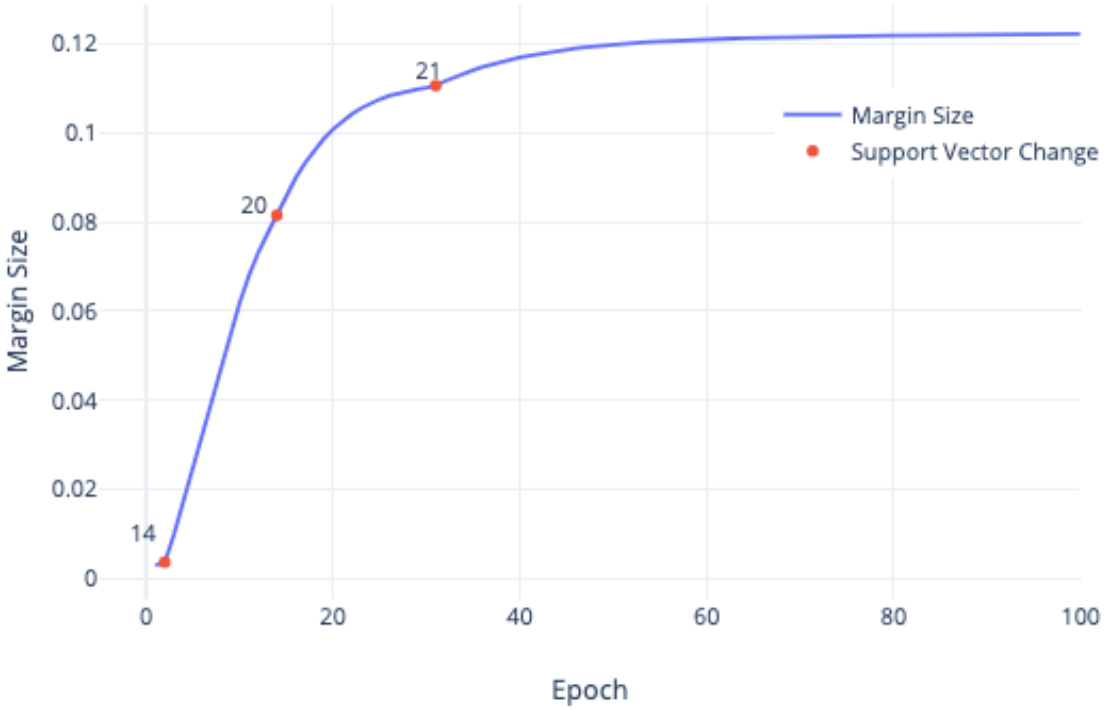

Figure 1: **Monotonic increase of the margin** The growth of the margin $\min_n y_n f(V; x_n)$ in the binary case. The network converges to 100% training accuracy in epoch 3 and changes the support vectors three times (red dots – the numbers above them correspond to the index of the datapoint that becomes the support vector) before stabilizing. As predicted by theory, the margin is non-decreasing. As the support vectors change, notice that the rate of margin growth accelerates.

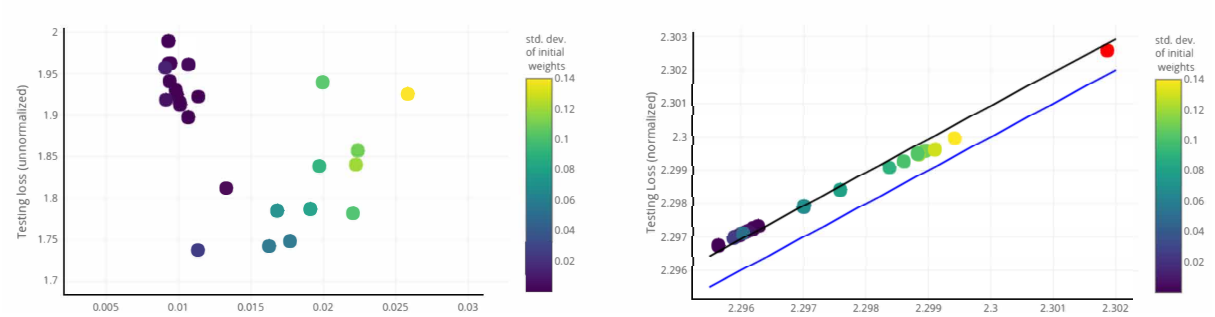

Figure 2: *Empirical evidence of generalization by normalized networks with respect to the cross entropy loss. The left graph shows testing vs training cross-entropy loss for networks each trained on the same data sets (CIFAR10) but with different initializations, yielding zero classification error on training set but different testing errors. The right graph shows the same data, that is testing vs training loss for the same networks, now normalized by dividing each weight by the Frobenius norm of its layer. Notice that all points have zero classification error at training. The red point on the top right refers to a network trained on the same CIFAR10 data set but with randomized labels. It shows zero classification error at training and test error at chance level. The top line is a square-loss regression of slope* 1 *with positive intercept. The bottom line is the diagonal at which training and test loss are equal. The networks are 3-layer convolutional networks. The left can be considered as a visualization of generalization bounds when the Rademacher complexity is not controlled. The right hand side is a visualization of the same relation for normalized networks that is* $L(f) \leq \hat{L}(f) + c_1 \mathbb{R}_N(\tilde{\mathbb{F}}) + c_2 \sqrt{\ln(\frac{1}{\delta})/2N}$. *Under our conditions for* N *and for the architecture of the network the terms* $c_1 \mathbb{R}_N(\tilde{\mathbb{F}}) + c_2 \sqrt{\ln(\frac{1}{\delta})/2N}$ *represent a small offset.*

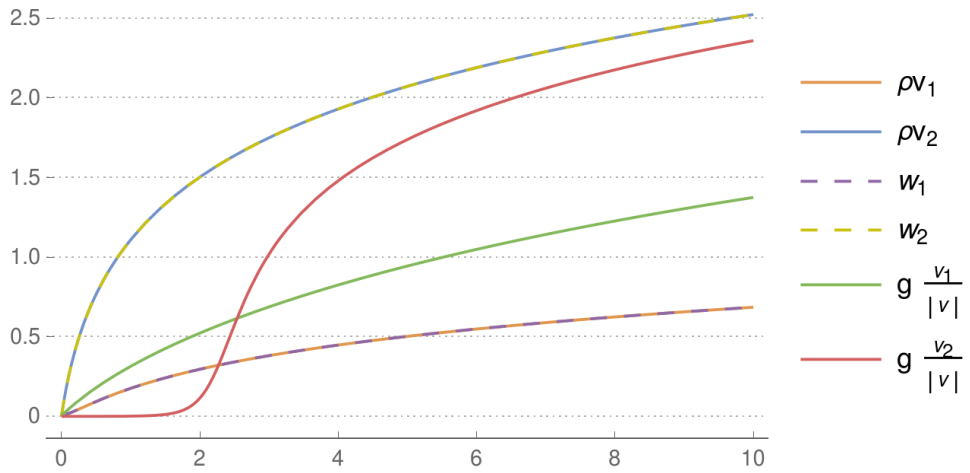

Figure 3: *Example comparison of dynamics* $\dot{W}_k = -\frac{\partial L}{\partial W_k}$ *(dashed lines) and their equivalence to the reparametrization of the unconstrained dynamics (orange and blue) as a function of time. Standard weight normalization leads to different trajectories of gradient descent (in green and red). The example is that of a linear network with only two parameters and two training examples, using an exponential loss.*

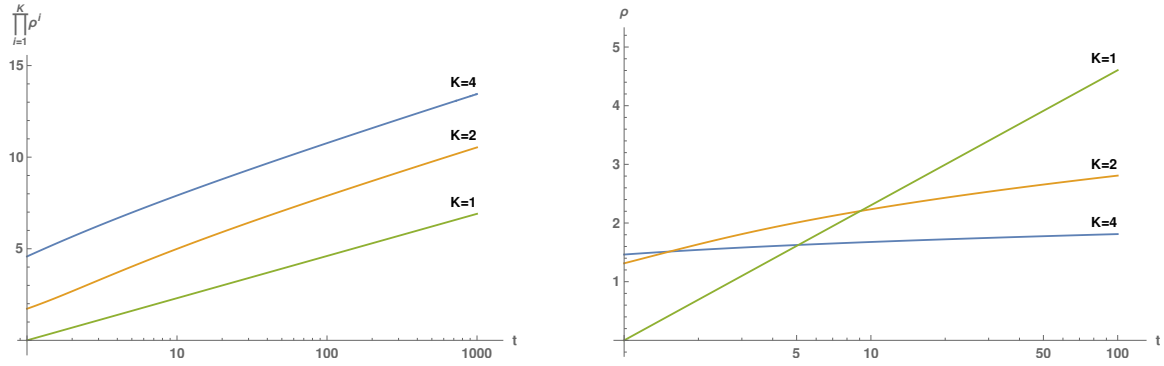

Figure 4: *The left graph shows how the product of weights* $\prod_{i=1}^{K}$ *scales as the number of layers grows when running gradient descent with an exponential loss. In the 1-layer case we have* $\rho = \rho \sim \log t$*, whereas for deeper networks the product of norms grows faster than logarithmically. As we increase the number of layers, the individual weights at each layer diverge slower than in the 1-layer case, as seen on the right graph.*

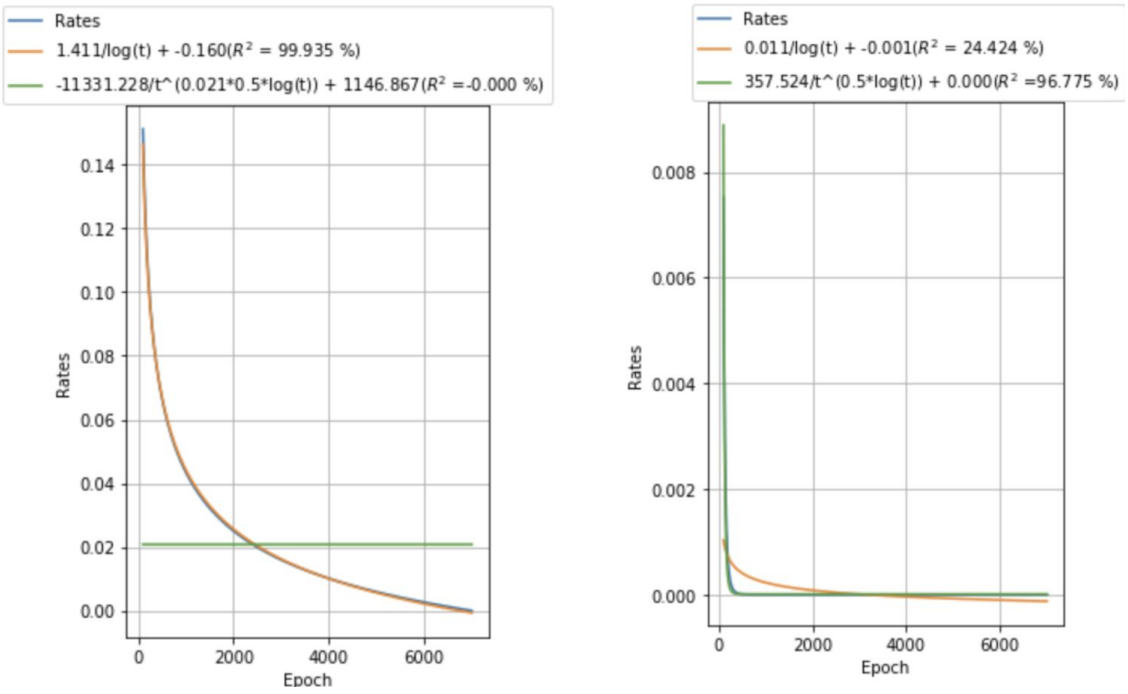

Figure 5: *Rates of convergence of $v(t)$ for a linear network trained on 1 training example on the IRIS dataset. Left: Standard gradient descent converges in direction at the rate $\mathcal{O}(\frac{1}{\log t})$. Right: Weight Normalization changes the convergence rate to that of $\mathcal{O}(t^{-1/2 \log t})$. We obtain more complex behavior in the presence of multiple support vectors with different margins.*

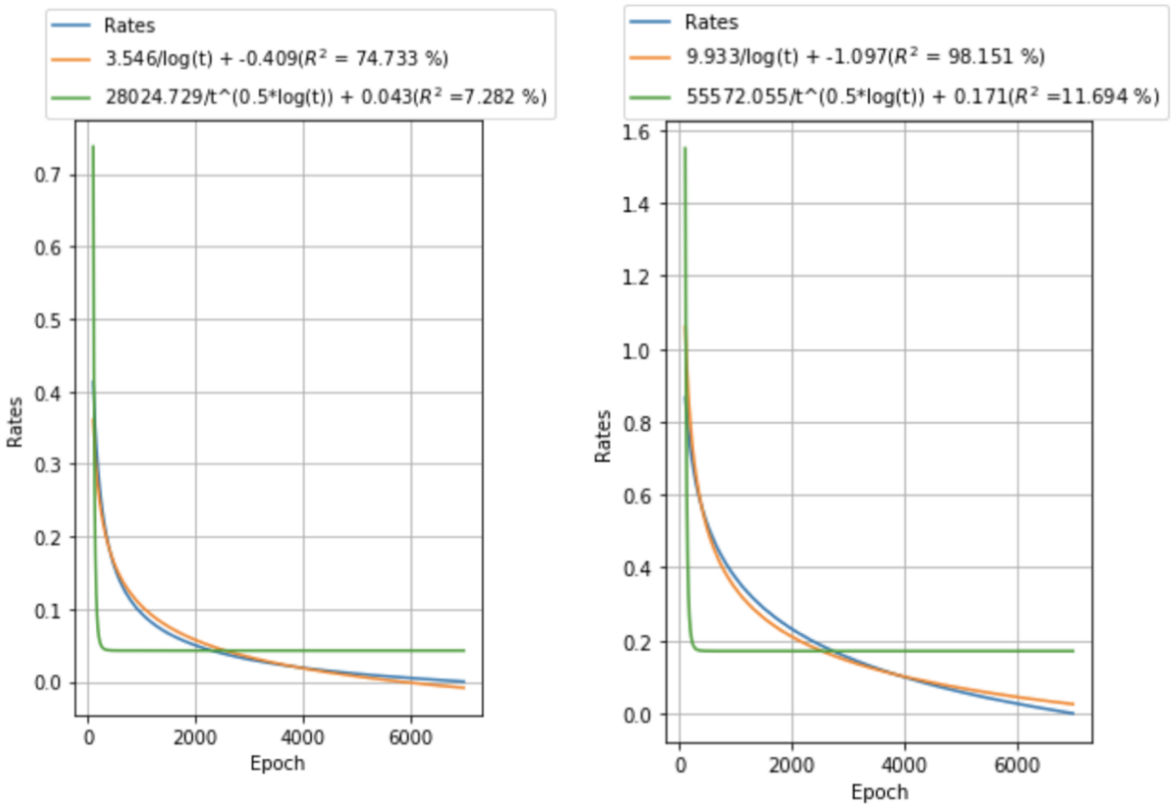

Figure 6: *Rates of convergence of $v(t)$ for a linear network trained on all the training examples on the IRIS dataset. Left: Standard gradient descent. Right: Weight Normalization. The goodness of fit in the both cases drops compared to the 1 example case. In the case of WN, $\mathcal{O}(\frac{1}{\log t})$ seems like a much better fit. Multiple data points imply more complicated dynamics and a slower convergence to the support on a small subset of inputs.*

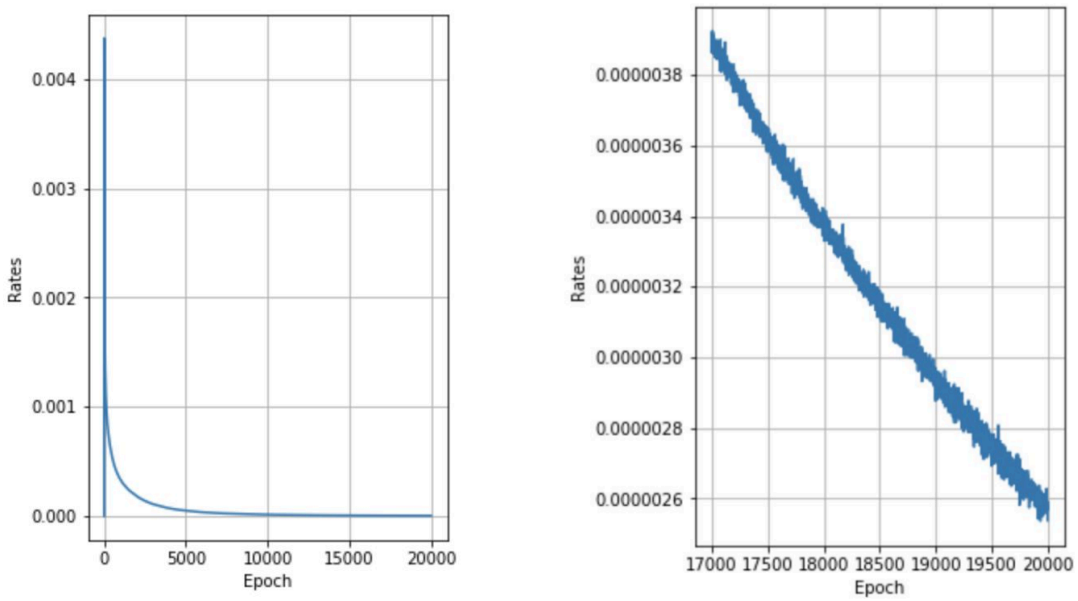

Figure 7: *Rates of convergence of $v(t)$ compared at every epoch for a linear network trained on all the training examples on the IRIS dataset. Left: All Epochs. Right: Last 3000 epochs. Towards convergence, there are no oscillations in the rates. The oscillating behavior observed in the last epochs are a result of numerical error.*

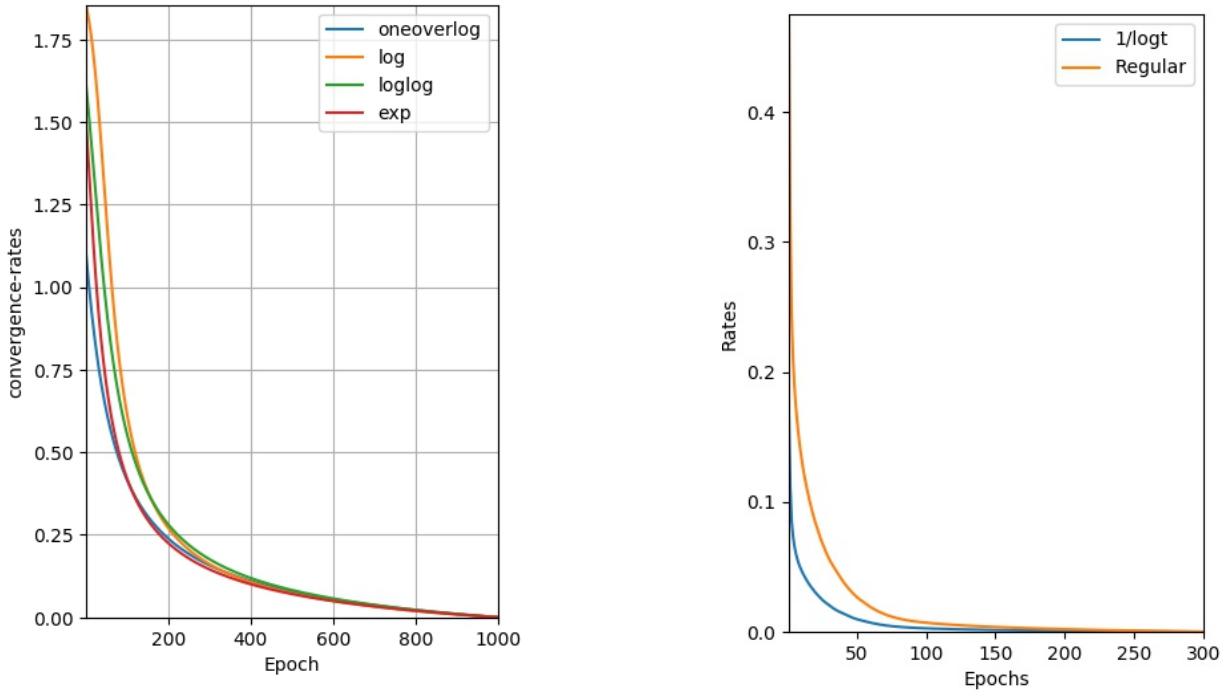

Figure 8: *Rates of convergence of $v(t)$ with ρ set as several functions of t for a linear network trained on all the training examples on the IRIS dataset. Left: Convergence rates when ρ was set to different functions of t. Right: Rates of convergence with rho set to $\frac{1}{\log t}$ compared to the original rates of convergence.*

.

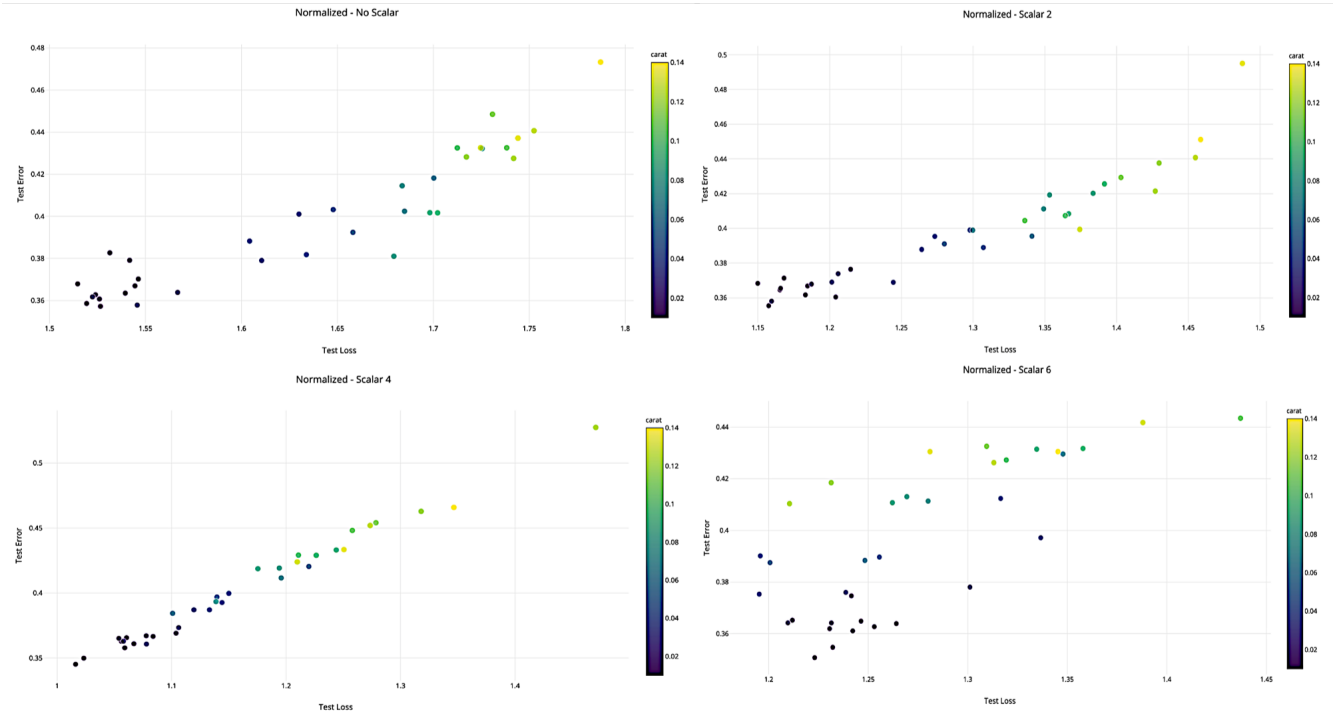

Figure 9: *The experiments are measured as in Figure 2, here we plot the normalized test loss vs. test error. As we scale the normalized network, the linear relationship disappears around* $\rho \approx 4$.

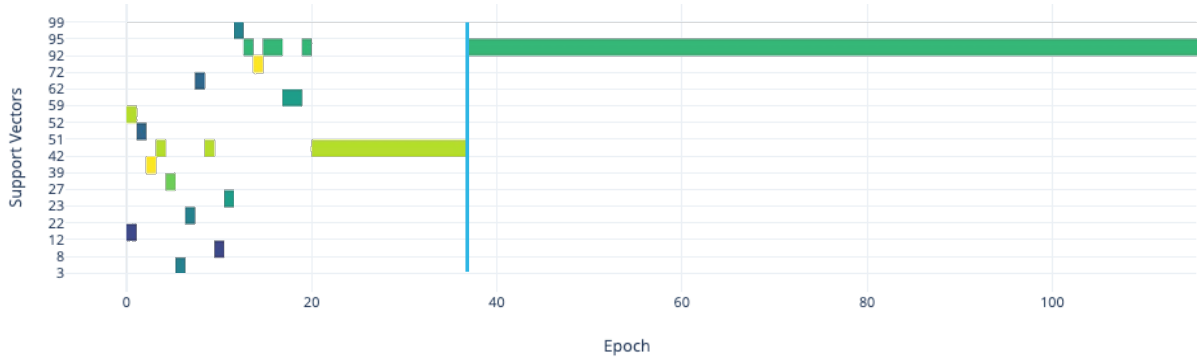

Figure 10: *A 6-layer neural network implemented in PyTorch was trained on a subset of 100 datapoints from CIFAR-10 with Gradient Descent (GD) on cross-entropy loss. Support vectors were obtained by taking the smallest of the distances between the class output and the second highest class output for each datapoint,* $\operatorname{argmin}(f_{y_i} - \max_{j \neq i} f_{y_j})$. *The blue line represents data separation (100 % accuracy training). The network has 4 convolutional layers (filter size 3 × 3, stride 2) and two fully-connected layers. Each convolutional layer is followed by batch-normalization. The number of feature maps (i.e., channels) in hidden layers are 16, 32, 64, 128 respectively. In total, the network has 6,400 parameters. The learning rate was constant throughout and set to 0.1 and momentum to 0.9. Neither data augmentation nor regularization is performed.*

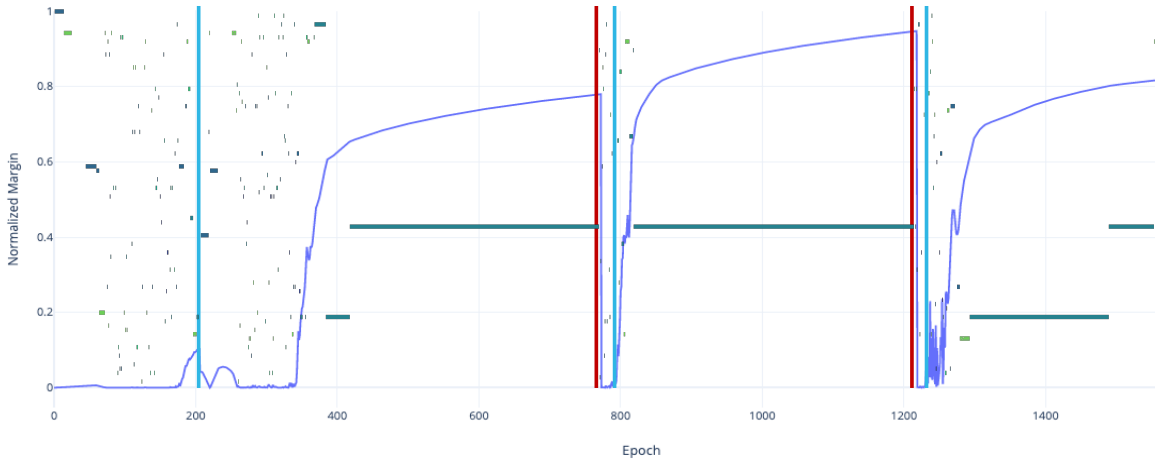

Figure 11: *A convolutional neural network with same parameters and similar architecture as the one in Figure 9, with the distinction of no batch-normalization, was trained on 100 data points from the CIFAR10 dataset. The margin* $\operatorname{argmin}(f_{y_i} - \max_{j \neq i} f_{y_j})$ *and the data points closest to the margin solution (support vectors) at each epoch are shown. The network separates the data (reaches* 100% *training accuracy) at epoch 200 represented by the first blue line. After the rate of change of the margin is* ≤ 0.00002 *(margin convergence) perturbations to the network were performed with respect to the gradient of each individual weight,* $0.5 \|\nabla_w L\|$*. Each perturbation was performed after the network separates the data again (blue lines) and reaches margin convergence as indicated by the red lines. The network converges to one support vector and then jumps between minima after each perturbation but eventually comes back to the same support vector.*