Complexity control by gradient descent in deep networks

Tomaso Poggio1✉, Qianli Liao1 & Andrzej Banburski1

Overparametrized deep networks predict well, despite the lack of an explicit complexity control during training, such as an explicit regularization term. For exponential-type loss functions, we solve this puzzle by showing an effective regularization effect of gradient descent in terms of the normalized weights that are relevant for classification.

1Center for Brains, Minds, and Machines, MIT, Cambridge, Massachusetts, USA. ✉email: tp@ai.mit.edu
Once upon a time, models needed more data than parameters to provide a meaningful fitting. Deep networks seem to avoid this basic constraint. In fact, more weights than data is the standard situation for deep-learning networks that typically fit the data and still show good predictive performance on new data. Of course, it has been known for some time that the key to good predictive performance is controlling the complexity of the network and not simply the raw number of its parameters. The complexity of the network depends on appropriate measures of complexity of the space of functions realized by the network such as VC dimension, covering numbers and Rademacher numbers. Complexity can be controlled during optimization by imposing a constraint, often under the form of a regularization penalty, on the norm of the weights, as all the notions of complexity listed above depend on it. The problem is that there is no obvious control of complexity in the training of deep networks! This has given an aura of magic to deep learning that there is no obvious control of complexity in the training of deep networks. The basic regularization in typical training of deep networks is the one per layer, with matching dimensions. The symbol $W$ is used to denote the set of $W$ matrices $k = 1, \ldots, K$. 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 $W^K \in \mathbb{R}^{1 \times K_i}$. As mentioned, 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 from the input layer where the bias is instantiated by one of the input dimensions being a constant. The activation function is the Rectified Linear Unit (ReLU) activation. For ReLU activations, the following important positive one-homogeneity property holds $\sigma(z) = \frac{\partial \sigma(z)}{\partial z} z$. For the network, homogeneity implies $f_{W}(x) = \prod_{k=1}^{K} \rho_k f_{V_k}(x_{1}; \ldots; x_{K}; x_n)$, where $W_k = \rho_k V_k$. The network is a function $f(x) = f(W_1; \ldots; W_K; x)$ where $x$ is the input and the weight matrices $W_k$ are the parameters. We define the normalized network $f_{\hat{x}}$ as $f_{\hat{x}} = f_{\rho} f_{\hat{x}}$, with $\|V_1\| = 1$, $\rho = \prod_{k=1}^{K} \rho_k$, $\hat{z}$ is the associated class of “normalized neural networks" $f(x)$. It is noteworthy that the definitions of $\rho_k$, $\hat{z}$ and $\hat{f}$ all depend on the choice of the norm used in normalization. It is also worth noting that because of homogeneity of the ReLU network $f(x) = f_{\hat{x}}(x)$, the signs of $f$ and $\hat{f}$ are the same. For simplicity of notation we consider for each weight matrix $V_k$ the corresponding “vectorized" representation in terms of vectors $W_k^{ij} = W_k$ for each $k$ layer.

We use the following definitions and properties (for a vector $w$ and the 2-norm) neglecting indexes:

- **Define** $\frac{\partial w}{\partial w} = w$; thus $w = \left\| w \right\|_2 v$ with $\left\| v \right\|_2 = 1$.
- The following relations are easy to check:
  1. $\frac{\partial \left\| w \right\|_2}{\partial w} = v$; 
  2. $S = I - vv^T = I - \frac{w}{\left\| w \right\|_2} w^T$; 
  3. $\frac{\partial v}{\partial w} = S / \left\| w \right\|_2$.

### Results

**Loss functions.** We consider for simplicity of exposition the case of binary classification. We call “loss" the measure of performance of the network $f$ on a training set $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$. The most common loss optimized during training for binary classification is the logistic loss $L(f) = \frac{1}{N} \sum_{n=1}^{N} \ln(1 + e^{-y_nf(x_n)})$. We focus on the closely related, simpler exponential loss $L(f(w)) = \sum_{n=1}^{N} e^{-y_nf(x_n)}$. We call classification “error" $\frac{1}{N} \sum_{n=1}^{N} H(-y_nf(x_n))$, where $y$ is binary ($y \in \{-1, +1\}$) and $H$ is the Heaviside function with $H(yf(x)) = 1$ if $-yf(x) > 0$, which correspond to the wrong classification. We say that $f$ separates the data if $y_nf(x_n) > 0, \forall n$. We will typically assume that GD at some $t > T_0$ will reach an $\hat{f}$ that separates the data (which is usually the case for overparametrized deep networks). There is a close relation between the exponential or logistic loss and the classification error: both the exponential and the logistic losses are upper bounds for the classification error. Minimizing the exponential or logistic loss implies minimizing the classification error. Minimization of the loss can be performed by GD techniques. In today’s praxis, stochastic GD (SGD) is used to train deep networks. We focus here on GD for simplicity. Our main results should also hold for SGD.

**Deep networks.** We define a deep network with $K$ layers and coordinate-wise scalar activation functions $\sigma(z) : \mathbb{R} \rightarrow \mathbb{R}$ as the set of functions $f(W; x) = \sigma(W^K a(W^{K-1} \cdots \sigma(W^1)x))$, where the input is $x \in \mathbb{R}^d$, the weights are given by the matrices $W_k$, one per layer, with matching dimensions. The symbol $W$ is used to denote the set of $W$ matrices $k = 1, \ldots, K$. 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 $W^K \in \mathbb{R}^{1 \times K_i}$. As mentioned, 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 from the input layer where the bias is instantiated by one of the input dimensions being a constant. The activation function is the Rectified Linear Unit (ReLU) activation. For ReLU activations, the following important positive one-homogeneity property holds $\sigma(z) = \frac{\partial \sigma(z)}{\partial z} z$. For the network, homogeneity implies $f_{W}(x) = \prod_{k=1}^{K} \rho_k f_{V_k}(x_{1}; \ldots; x_{K}; x_n)$, where $W_k = \rho_k V_k$. The network is a function $f(x) = f(W_1; \ldots; W_K; x)$ where $x$ is the input and the weight matrices $W_k$ are the parameters. We define the normalized network $f_{\hat{x}}$ as $f_{\rho} f_{\hat{x}}$, with $\|V_1\| = 1$, $\rho = \prod_{k=1}^{K} \rho_k$, $\hat{z}$ is the associated class of “normalized neural networks” $f(x)$. It is noteworthy that the definitions of $\rho_k$, $\hat{z}$ and $\hat{f}$ all depend on the choice of the norm used in normalization. It is also worth noting that because of homogeneity of the ReLU network $f(x) = f_{\hat{x}}(x)$, the signs of $f$ and $\hat{f}$ are the same. For simplicity of notation we consider for each weight matrix $V_k$ the corresponding “vectorized" representation in terms of vectors $W_k^{ij} = W_k$ for each $k$ layer.

We use the following definitions and properties (for a vector $w$ and the 2-norm) neglecting indexes:

- **Define** $\frac{\partial w}{\partial w} = w$; thus $w = \left\| w \right\|_2 v$ with $\left\| v \right\|_2 = 1$.
- The following relations are easy to check:
  1. $\frac{\partial \left\| w \right\|_2}{\partial w} = v$; 
  2. $S = I - vv^T = I - \frac{w}{\left\| w \right\|_2} w^T$; 
  3. $\frac{\partial v}{\partial w} = S / \left\| w \right\|_2$.

### Training by unconstrained gradient descent

Consider the typical training of deep networks. The exponential loss (more in general an exponential-type loss such as the cross-entropy) is
minimized by GD. The gradient dynamics is given by
\[ W_{k}^{j} = -\frac{\partial L}{\partial W_{k}^{j}} = \sum_{n=1}^{N} y_{n} \frac{\partial f(x_{n})}{\partial W_{k}^{j}} e^{-y_{n}f(x_{n})}. \]  

(2)

Clearly there is no explicit regularization or norm control in the typical GD dynamics of Eq. (2). Assuming that for \( T > T_{0} \) GD achieves separability, the empirical loss goes to zero and the norms of the weights \( \rho_{k} = \|W_{k}\| \) grow to infinity \( \forall k \). For classification, however, only the normalized network outputs matter because of the softmax operation.

Training by constrained gradient descent. Let us contrast the dynamics of Eq. (4) is the same as of the weight normalization technique are in common use for training deep networks. The time derivatives, the dynamics for \( \rho \) and \( V_{k} \), \( \rho_{k} \) and \( \dot{V}_{k} \) as the normalized network \( f(x) \) is what matters for classification, and other key properties such as the margin depend on it.

In terms of \( \rho, V_{k} \), the unconstrained gradient of \( L \) gives
\[ \dot{\rho}_{k} = -V_{k}^{T} \frac{\partial L}{\partial W_{k}} \quad \dot{V}_{k} = -\rho_{k} \frac{\partial L}{\partial W_{k}} \]  

(3)
as \( \dot{\rho}_{k} = -\frac{\partial L}{\partial \rho_{k}} = \frac{\partial W_{k}}{\partial \rho_{k}} \) and \( \dot{V}_{k} = -\frac{\partial L}{\partial \rho_{k}} = -\frac{\partial W_{k}}{\partial \rho_{k}} \).

There are several ways to enforce the unit norm constraint on the weight matrix at each layer (if \( \rho = 2 \)) is the Frobenius norm). It is noteworthy that the relevant function is not \( f(x) \) and the associated \( W_{k} \) but rather \( f_{i}(x) \), and the associated \( V_{k} \) as the normalized network \( f(x) \) with \( \rho_{k} \) and \( \dot{V}_{k} \).

The following dynamics for \( \rho_{k} = \|W_{k}\| \) and \( V_{k} \):
\[ \dot{\rho}_{k} = V_{k} \dot{W}_{k} \quad \dot{V}_{k} = \frac{S_{k}}{\rho_{k}} \dot{W}_{k} \]  

(6)

where \( S_{k} = I - V_{k}V_{k}^{T} \) emerges this time from the change of variables, as \( \frac{\partial L}{\partial \rho_{k}} = \frac{\partial L}{\partial \rho_{k}} \). Inspection of the equation for \( \dot{V}_{k} \) shows that there is a unit constraint on the \( L_{2} \) norm of \( V_{k} \), because of the presence of \( S \). In fact, a tangent gradient transformation on \( \dot{V}_{k} \) would not change the dynamics, as \( S \) is a projector and \( S^{2} = S \). Consistently with this conclusion, unconstrained GD has the same critical points for \( V_{k} \) as weight normalization but a somewhat different dynamics: in the one-layer case, weight normalization is
\[ \dot{\rho} = v^{T} \dot{w} \quad \dot{v} = S \rho \]  

(7)

The two dynamical systems are thus quite similar, differing by a \( \rho^{2} \) factor in the \( \dot{v} \) equations. It is clear that 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, as for any \( t > 0 \), \( \rho(t) > 0 \).

Importantly, the almost equivalence between constrained and unconstrained GD is true only when \( \rho = 2 \) in the unit norm constraint. In both cases the stationary point (for fixed but usually very large \( \rho \), that is a very long time) are the same and likely to be (local) minima. In the case of deep networks, we expect multiple such minima for any finite \( \rho \).

Convergence to minimum norm and maximum margin. Consider the flow of \( V_{k} \) in Eq. (6) for fixed \( \rho \). If we assume that for \( t > T_{0} \), \( f(V;x) \) separates the data, i.e., \( \forall n f_{i}(V;x_{n}) \neq 0 \), then \( \frac{\partial L}{\partial \rho_{k}} > 0 \), i.e., \( \rho \) diverges to \( \infty \) with \( \lim_{t \rightarrow \infty} \dot{\rho} = 0 \). In the one-layer network case, the dynamics yields \( \rho \approx \log \rho \) asymptotically. For deeper networks, this is different. Banburski et al. \( ^{5} \) shows that the product of weights at each layer diverges faster than logarithmically, but each individual layer diverges slower than in the one-layer case. Banburski et al. \( ^{5} \) also shows that \( \rho_{k}^{2} \) (the rate of growth of \( \|\rho_{k}\|^{2} \) is independent of \( k \).

The stationary points at which \( \dot{V}_{k} = 0 \) for fixed \( \rho \)—if they exist—satisfy the necessary condition for a minimum, i.e.,
\[ \sum_{n} e^{-\gamma \rho f_{i}(x_{n})} \left( \frac{\partial f_{i}(x_{n})}{\partial V_{k}^{T}} - V_{k}^{T} f_{i}(x_{n}) \right) = 0 \]  

(9)

are critical points of the loss for fixed \( \rho \) (as the domain is compact minima exist). Let us call them \( V_{k}(\rho) = \min_{\|V_{k}\| = 1} \)
\[ \lim_{\rho \rightarrow \infty} \min_{\|V_{k}\| = 1} \sum_{n} e^{-\gamma \rho f_{i}(x_{n})} = \min_{\rho \rightarrow \infty} \min_{\|V_{k}\| = 1} e^{-\rho f_{k}(x_{n})} \]  

(10)

where \( x^{*} \) corresponds to the training data that are support vectors and have the smallest margin. This provides a solution \( V_{k}(\infty) \), which corresponds to a maximum margin solution, because \( \lim_{\rho \rightarrow \infty} V_{k}(\rho) = \min_{\|V_{k}\| = 1} \max_{\|x\| \leq 1} f_{k}^{*}(x_{n}) \) (see also ref. \( ^{5} \)). Those maximum margin solutions are also minimum norm solution in terms of the \( W_{k} \) for a fixed margin \( \geq 1 \).

In summary, the dynamics of \( V_{k} \) for each fixed \( \rho \) converges to a critical point which is likely to be a minimum on the boundary of the \( p \) ball. For \( \rho \rightarrow \infty \), the stationary points of the full dynamical system Eq. (6) are reached. These points are degenerate equilibria.\( ^{5} \)
Discussion

The main conclusion of this analysis is that unconstrained GD in $W_k$ followed by normalization of $W_k$ is equivalent to imposing an $L_2$ unit norm constraint on $V_k = \frac{W_k}{\|W_k\|_2}$ during GD. For binary and multiclassification, only the normalized weights are needed. To provide some intuition, consider that GD is steepest descent wrt the $L_2$ norm and the steepest direction of the gradient depends on the norm. The fact that the direction of the weights converge to stationary points of the gradient under a constraint is the origin of the hidden complexity control, described as implicit bias of GD by Srebro and colleagues, who first showed its effect in the special case of linear networks under the exponential loss.

In addition to the approach summarized here, another elegant theory leading to several of the same results, has been developed around the notion of margin, which is closely related, as in the case of support vector machines, to minimization of an exponential-type loss function under a norm constraint.

In summary, there is an implicit regularization in deep non-linear networks trained on exponential-type loss functions, originating in the GD technique used for optimization. The solutions are in fact the same as those are obtained by vanishing regularized optimization. This is thus similar to—but more robust than—the classical implicit regularization induced by iterative GD on linear networks under the square loss and with appropriate initial conditions. In our case, the maximum margin solutions are independent of initial conditions and the linearity of the network. The specific solutions, however, are not unique, unlike the linear case: they depend on the trajectory of gradient flow, each corresponding to one of multiple minima of the loss, each one being a margin maximizer. In general, each solution will show a different test performance. Characterizing the conditions that lead to the best among the margin maximizers is an important open problem.

The classical analysis of ERM algorithms studies their asymptotic behavior for the number of data $n$ going to infinity. In this limiting classical 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 paper implies that there is asymptotic generalization and consistency in deep networks in the classical regime (see Fig. 1). However, as we mentioned, it has been shown in the case of kernel regression, that there are situations in which there is simultaneous interpolation of the training data and good expected error. This is a modern regime in which $D > n$ but $\gamma = \frac{D}{n}$ is constant. In the linear case, it corresponds to the limit for $\lambda = 0$ of regularization, i.e., the pseudoinverse. It is likely that deep nets may have a similar regime, in which case the implicit regularization described here is an important prerequisite for a full explanation—as it is the case for kernel machines under the square loss. In fact, the maximum margin solutions we characterize here are equivalent to minimum norm solutions (for margin equal to 1, see ref. 5). Minimum norm of the weight matrices implies minimum uniform stability and thus suggests minimum expected error, see ref. 11. This argument would explain why deep networks trained with exponential losses predict well and why classification error does not increase with overparametrization (see Fig. 2). It would also explain, in the case of kernel methods and square-loss regression, why the pseudoinverse solution provides good expected error and at the same time perfect interpolation on the training set with a data-dependent double-descent behavior.

As we mentioned, capacity control follows from convergence of the normalized weights during GD to a regularized solution with vanishing $\lambda$. It is very likely that the same result we obtained for GD also holds for SGD in deep networks (the equivalence holds for linear networks). The convergence of SGD usually follows convergence of GD, although rates being different. The Robbins–Siegmund theorem is a tool to establish almost sure convergence under surprisingly mild conditions.

It is not clear whether a similar effective regularization should also hold for deep networks with more than two layers trained with square loss. In fact, we have not been able to find a mechanism that could lead to a similarly robust regularization. Therefore, we conjecture that deep networks trained with square loss (with more than two layers and not reducible to kernels) do not converge to minimum norm solutions, unlike the same networks trained on exponential-type losses.

![Fig. 1 Classical generalization and consistency in deep networks.](https://doi.org/10.1038/s41467-020-14663-9)
The dynamics of weight normalization (Eq. (7)) vs. the dynamics of the unconstrained gradient (Eq. (8)). Under the same simplified assumptions on the training data, the weight normalization dynamics converges much faster— as $\frac{1}{\log t}$— than the typical dynamics, which converges to the stationary point as $\frac{t}{\log t}$. This prediction was verified with simulations. Together with the observation that $\rho(t)$ associated with Eq. (8) is monotonic in $t$ after separability is reached, it suggests exploring a family of algorithms that consist of an independently driven forcing term $\rho(t)$ coupled with the equation in $V_k$ from Eq. (8).

$$V_k = \frac{\rho}{\rho_k} \sum_{n=1}^{N_k} e^{\frac{-\rho}{\rho_k}} \left( \frac{\partial f_{V_k}(x_n)}{\partial V_k} - V_k f_{V_k}(x_n) \right).$$  \hspace{1cm} (11)

The open question is of course what is the optimal $\rho(t)$ schedule for converging to the margin maximizer that is best in terms of expected error.

**Data availability**
All relevant data are publicly available (Github: https://github.com/liaoq/natcom2020).

**Code availability**
All relevant code are publicly available (Github: https://github.com/liaoq/natcom2020).

Received: 14 September 2019; Accepted: 22 January 2020; Published online: 24 February 2020

**References**
1. Zhang, C., Bengio, S., Hardt, M., Recht, B. & Vinyals, O. Understanding deep learning requires rethinking generalization. In International Conference on Learning Representations (ICLR) (Toulon, France, 2017).
2. Rosasco, L. & Villa, S. Learning with incremental iterative regularization. In Advances in Neural Information Processing Systems, 1630–1638 (2015).
3. Engl, H., Hanke, M. & Neubauer, A. Regularization of Inverse Problems (Kluwer Academic Publishers, 2000).
4. Douglas, S. C., Amari, S. & Kung, S. Y. On gradient adaptation with unit-norm constraints. IEEE Transact. Signal Process. 48(June), 1843–1847 (2000).
5. Banburski, A. et al. Theory of deep learning III: dynamics and generalization in deep networks. CBMM Memo No. 090 (2019).
6. Salimans, T. & Kingma, D. P. Weight normalization: a simple reparameterization to accelerate training of deep neural networks. In Advances in Neural Information Processing Systems arXiv:1602.07868v3 (Barcelona, Spain, 2016).
7. Soudry, D., Hoffer, E. & Srebro, N. The implicit bias of gradient descent on separable data. J. Mach. Learn. Res. 19, 701–57 (2017).
8. Nacson, M. S., Gunasekar, S., Lee, J. D., Srebro, N. & Soudry, D. Lexicographic and depth-sensitive margins in homogeneous and non-homogeneous deep models. In Proceedings of the 36th International Conference on Machine Learning (ICML) (Long Beach, California, 2019).
9. Lyu, K. & Li, J. Gradient descent maximizes the margin of homogeneous neural networks. CoRR, abs/1906.05890 (2019).
10. Neyshabur, B., Tomioka, R., Salakhutdinov, R. & Srebro, N. Geometry of optimization and implicit regularization in deep learning. CoRR, abs/1705.03071 (2017).
11. Poggio, T., Kur, G. & Banburski, A. Double descent in the condition number. CBMM memo 102 (2019).
12. Liang, T. & Rakhlin, A. Just interpolate: kernel “Ridgeless” regression can generalize. In The Annals of Statistics. (2019).
13. Rakhlin, A. & Zhou, X. Consistency of interpolation with laplace kernels is a high-dimensional phenomenon. In Proceedings of the Thirty-Second Conference on Learning Theory (COLT), PMLR 99, 2595–2623, (2019).
14. Nacson, M. S., Srebro, N. & Soudry, D. Stochastic gradient descent on separable data: exact convergence with a fixed learning rate. In AISTATS 3051–3059 (2019).

**Acknowledgements**
This material is based upon work supported by the Center for Minds, Brains 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.

**Author contributions**
T.P. developed the basic theory. Q.L. and A.B. run simulations. All three authors wrote the paper.

**Competing interests**
The authors declare no competing interests.

**Additional information**
Supplementary information is available for this paper at https://doi.org/10.1038/s41467-020-14663-9.

**Correspondence** and requests for materials should be addressed to T.P.

**Peer review information** Nature Communications thanks the anonymous reviewers for their contribution to the peer review of this work. Peer reviewer reports are available.

**Reprints and permission information** is available at http://www.nature.com/reprints

**Publisher’s note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

---

Fig. 2 No overfitting in deep networks. Empirical and expected error in CIFAR-10 as a function of number of neurons in a 5-layer convolutional network. The expected classification error does not increase when increasing the number of parameters beyond the size of the training set.