Weight and Batch Normalization implement Classical Generalization Bounds

Andrzej Banburski [1] **Qianli Liao** [1] **Brando Miranda** [1] **Lorenzo Rosasco** [1] **Jack Hidary** [2] **Tomaso Poggio** [1]

Abstract

Classical generalization bounds for classification suggest maximization of the margin of a deep network under the constraint of unit Frobenius norm of the weight matrix at each layer. We show that (1) this goal can be achieved by gradient algorithms enforcing a unit norm constraint; (2) weight normalization and batch normalization enforce a unit norm constraint. This provides a fundamental reason for their effectiveness in terms of test error.

1. Introduction

A satisfactory theoretical characterization of deep learning should answer questions about 1) *representation power* of deep networks 2) *optimization* of the empirical risk 3) *generalization properties* of gradient descent techniques. This paper addresses the third question: 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 (Hardt & Ma, 2016; Neyshabur et al., 2017; Sokolic et al., 2017; Bartlett et al., 2017; Zhang et al., 2017)).

The results described here are:

- generalization can be achieved by gradient descent algorithms that enforce a weight normalization constraint;
- the fundamental reason for the effectiveness of existing weight normalization and batch normalization gradient descent techniques is that they implement maximization of the margin under unit norm constraint.

[1]Center for Brains, Minds and Machines, MIT [2]Alphabet (Google) X. Correspondence to: Andrzej Banburski <kappa666@mit.edu>, Tomaso Poggio <tp@ai.mit.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

2. Deep networks: definitions and properties

Definitions We define a deep network with K layers with the usual coordinate-wise scalar activation functions $\sigma(z) : \mathbb{R} \to \mathbb{R}$ as the set of functions $f(W;x) = \sigma(W^K\sigma(W^{K-1}\cdots\sigma(W^1x)))$, where the input is $x \in \mathbb{R}^d$, 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 $W^K \in \mathbb{R}^{1,K_l}$. 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 is the ReLU activation.

Positive one-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 $f(W;x) = \prod_{k=1}^{K}\rho_k\tilde{f}(V_1, \cdots, V_K; x_n)$, where $W_k = \rho_k V_k$ with the Frobenius norm $||V_k|| = 1$ (for convenience). 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 $\rho = \rho_1 \dots \rho_K$, $\mathbb{F}$ is the class of neural networks described above and accordingly $\tilde{\mathbb{F}}$ is the corresponding class of normalized neural networks. This invariance property of the function f under transformations of W_k that leave the product norm the same is typical of ReLU (and linear) networks. In the paper we will refer to the norm of f meaning the product $\rho = \prod_{k=1}^{K}\rho_k$ of the Frobenius norms of the K weight matrices of f. Thus $f = \rho\tilde{f}$. Note that

$$\frac{\partial f}{\partial \rho_k} = \frac{\rho}{\rho_k}\tilde{f}. \tag{2}$$

Separable data When $y_n f(x_n) > 0\ \forall n = 1, \cdots, N$ we say that the data are separable that is they can all be correctly classified by the network f. Notice that this is a strong condition if f is linear but it can be often satisfied by overparametrized deep networks. In the following sections *we assume separable data.*

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 (Jin et al., 2017; Ge et al., 2015; Lee et al., 2016; Du et al., 2018c) 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 (Tian, 2017; Soltanolkotabi et al., 2019; Li & Yuan, 2017; Brutzkus & Globerson, 2017; Du et al., 2018a;b). Some local convergence results have also been proved (Zhong et al., 2017b;a; Zhang et al., 2018). The most interesting such approach is (Du et al., 2018b), which focuses on minimizing the training loss and proving that randomly initialized gradient descent can achieve zero training loss (see also (Li & Liang, 2018; Du et al., 2019; Zou et al., 2018)).

For generalization, which is the topic of this paper, existing work demonstrate that gradient descent works under the same situations as kernel methods and random feature methods (Daniely, 2017; Allen-Zhu et al., 2018; Arora et al., 2019). Closest to our approach – which is focused on the role of batch and weight normalization – is the paper (Wei et al., 2018). 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, in fact 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 implicitly controls the complexity.

4. Generalization Bounds and Implications

A typical margin bound for classification (Shawe-Taylor & Cristianini, 2004) is as follows

$$L_{binary}(f) \leq L_{surr}(f) + b_1 \frac{\mathbb{R}_N(\mathbb{F})}{\eta} + b_2 \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}} \quad (3)$$

where η is the margin, $L_{binary}(f)$ is the expected classification error, $L_{surr}(f)$ is the empirical loss of a surrogate loss such as the logistic or the exponential. For a point x, the margin is $\eta \sim y\rho\tilde{f}(x)$. Since $\mathbb{R}_N(\mathbb{F}) \sim \rho\mathbb{R}_N(\tilde{\mathbb{F}})$, the margin bound is optimized by effectively maximizing $\tilde{f}$ on the "support vectors" – that is the x_i, y_i s.t $\arg\min_n y_n \tilde{f}(x_n)$.

One can show (Banburski et al., 2019) that for separable data, the following holds true:

Theorem (informal) *Maximizing the margin subject to unit norm constraint is equivalent to minimize the norm of f subject to the constraint that the margin is greater than a positive constant.*

5. Optimization under norm constraint

The generalization bounds in the previous section imply maximization of the margin subject to the product of the layer norms being equal to one:

$$\arg\max_{\prod_k ||V_k||=1} \min_n y_n \rho \tilde{f}(x_n). \quad (4)$$

In words: *find the network weights that maximize the margin subject to a norm constraint*. The latter ensures a bounded Rademacher complexity and together they minimize the term $\frac{\mathbb{R}_N(\mathbb{F})}{\eta}$. In fact, existing generalization bounds such as Equation 6 in (Golowich et al., 2017), see also (Bartlett et al., 2017) are given in terms of products of upper bounds on the norm of each layer: the bounds require that each layer is bounded, rather than just the product is bounded.

This constraint is implied by a unit constraint on the norm of each layer, which defines an equivalence class of networks f because of Eq. (1).

In this section we focus on the classification case involving minimization of an exponential loss function

$$L(f(w)) = \sum_{n=1}^{N} e^{-f(W;x_n)y_n} = \sum_{n=1}^{N} e^{-\rho\tilde{f}(V;x_n)y_n} \quad (5)$$

with $||V_k||^2 = \sum_{i,j}(V_k)_{i,j}^2 = 1, \forall k$, that is under a unit norm constraint for the weight matrix at each layer. Clearly these constraints imply the constraint on the product of weight matrices in (4). There are several ways to implement the minimization in the tangent space of $||V||^2 = 1$ (see (Douglas et al., 2000)).

5.1. Gradient techniques for norm control

A review of gradient-based algorithms with unit-norm constraints (Douglas et al., 2000) 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 (Douglas et al., 2000). We assume, as usual, separable data. The four techniques are closely related and have the same goal: performing gradient descent optimization with a unit norm constraint. In the following we describe one of the techniques, the tangent gradient method.

5.2. Tangent gradient method

The method is based on the following statement:

Theorem (Douglas et al., 2000) *Let $||u||$ denote any 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*

$$\dot{u} = h_g(t) = Sg(t) = (I - \frac{uu^T}{|u|^2})g(t) \tag{6}$$

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

In particular, a form for g is $g(t) = \mu(t)\nabla_u L$, which is the gradient update in a standard gradient descent algorithm. We call $Sg(t)$ the tangent gradient transformation of g. Consider

$$\dot{W}_k^{i,j} = -\frac{\partial L}{\partial W_k^{i,j}} = \sum_{n=1}^{N} y_n \frac{\partial f(x_n; w)}{\partial W_k^{i,j}} e^{-y_n f(x_n;W)} \tag{7}$$

and let us assume that starting at some time t, $\rho(t)$– in $w = \rho\tilde{w}$ – is large enough that the following asymptotic expansion (as $\rho \to \infty$) is a good approximation: $\sum_n e^{-\rho(t)\tilde{f}(x_n)} \sim C \max_n e^{-\rho(t)\tilde{f}(x_n)}$, where C is the multiplicity of the minimal $\tilde{f}$.

The data points with the corresponding minimum value of the margin $y_n\tilde{f}(x_n)$ are the support vectors. They are a subset of cardinality C of the N datapoints, all with the same margin η. In particular, the term $g(t) = \rho(t)\sum_n e^{-\rho(t)\tilde{f}(x_n)}\frac{\partial \tilde{f}(x_n)}{\partial V_k}$ becomes $g(t) \approx \rho(t)e^{-\rho(t)\eta}\sum_{i=1}^{C}\frac{\partial \tilde{f}(x_i)}{\partial V_k}$.

A rigorous proof of the argument above can be regarded as an extension of the main theorem in (Rosset et al., 2003) from the case of linear functions to the case of one-homogeneous functions (it is easy to check that the proofs in (Rosset et al., 2003) hold for homogeneous networks and not only for linear ones). In fact (Wei et al., 2018) has theorems including such an extension.

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 (Poggio & Liao, 2017) *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 validity 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). In summary

Theorem*(informal statement)*

Under the assumption of separable data and convergence, the gradient methods with norm constraint converge to maximum margin solutions with unit norm.

5.3. Weight Normalization and Batch Normalization

Here we show that both techniques are gradient methods with norm constraint.

Classical weight normalization (Salimans & Kingm, 2016) 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:

$$\dot{g} = \frac{v}{||v||}\frac{\partial L}{\partial w} \tag{8}$$

and

$$\dot{v} = \frac{g}{||v||}S\frac{\partial L}{\partial w} \tag{9}$$

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

Batch normalization (Ioffe & Szegedy, 2015) for unit i normalizes the input 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 $\circ(\sigma_B^2 + \epsilon)^{-1/2}$. 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 the 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 the respect to the new variables are directly computed on its output. What is needed in the BN algorithm are only the derivative of L wrt the old variables as a function of the derivatives of L wrt new variables in order to use the chain rule to update the parameters below the BN module. 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 (Ioffe & Szegedy, 2015)). 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 transformation $\tilde{w} = \frac{w}{|w|}$. In a similar way the dynamics of $w = \frac{\partial L}{\partial w}$ induces the following dynamics on $\hat{X}$:

$$\dot{\hat{X}} = \frac{\partial \hat{X}}{\partial X} \dot{X} \tag{10}$$

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

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

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

Remarks

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

   $$||x||, ||\sigma(W_1 x)||, ||\sigma(W_2 \sigma(W_1 x))||, \cdots \tag{11}$$

   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 $||V||^2 = \sum_i ||V_i||_i^2$.

The main results of this section can be summarized as follows.

- *multilayer, nonlinear, deep networks trained with gradient descent methods with norm constraint (GDNC) converge to maximum margin solutions and therefore generalize because of classical classification bounds;*
- *popular gradient descent techniques such as weight normalization and batch normalization also generalize because they belong to the class of GDNC methods;*

This is similar to the situation for linear networks and in fact can be considered an extension of the result in (Soudry et al., 2017).

It is useful to emphasize that despite the similarities between the various GDNC methods they correspond to different dynamical flows.Furthermore our analysis has been restricted to the continuous case; the discrete case is expected to yield even greater differences.

6. Discussion

An implication of our results is that *the effectiveness of weight normalization and especially batch normalization is based on the fundamental reason of controlling the norm*. In the case of batch normalization, the reason is thus deeper than the original motivation of reducing covariate shifts. Controlling the norm of the weights during minimization is exactly what the generalization bounds prescribe.

This paper leaves a number of open problems. Does the empirical landscape have multiple global minima with different minimum norms, as we suspect? Why are local minima not "seen" by stochastic gradient descent? What is the explanation of the empirical observation that batch normalization is better than weight normalization? Is it possible to formulate generalization bounds that are quantitatively meaningful? The community has found answers (e.g. see (Banburski et al., 2019)) to some of the questions – such as why does standard gradient descent also generalize – but not to others.

Acknowledgements

We thank Yuan Yao, Misha Belkin, Jason Lee and especially Sasha Rakhlin for illuminating discussions. We would also like to thank the reviewer for the useful comments. 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

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. *CoRR*, abs/1811.04918, 2018. URL http://arxiv.org/abs/1811.04918.

Arora, S., Du, S. S., Hu, W., yuan Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *CoRR*, abs/1901.08584, 2019.

Banburski, A., Liao, Q., Miranda, B., Rosasco, L., Liang, B., Hidary, J., and Poggio, T. Theory iii: Dynamics and generalization in deep networks. abs/1903.04991, 2019. URL https://arxiv.org/abs/1903.04991.

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

Brutzkus, A. and Globerson, A. 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*, pp. 605–614, 2017. URL http://proceedings.mlr.press/v70/brutzkus17a.html.

Daniely, A. Sgd learns the conjugate kernel class of the network. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 2422–2430. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/6836-sgd-learns-the-conjugate-kernel-class-of-the-network.pdf.

Douglas, S. C., Amari, S., and Kung, S. Y. On gradient adaptation with unit-norm constraints. *IEEE Transactions on Signal Processing*, 48(6):1843–1847, June 2000. ISSN 1053-587X. doi: 10.1109/78.845952.

Du, S., Lee, J., Tian, Y., Singh, A., and Poczos, B. Gradient descent learns one-hidden-layer CNN: Don't be afraid of spurious local minima. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1339–1348, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018a. PMLR. URL http://proceedings.mlr.press/v80/du18b.html.

Du, S. S., Lee, J. D., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. *CoRR*, abs/1811.03804, 2018b. URL http://arxiv.org/abs/1811.03804.

Du, S. S., Lee, J. D., and Tian, Y. When is a convolutional filter easy to learn? In *International Conference on Learning Representations*, 2018c. URL https://openreview.net/forum?id=SkA-IE06W.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1eK3i09YQ.

Ge, R., Huang, F., Jin, C., and Yuan, Y. Escaping from saddle points - online stochastic gradient for tensor decomposition. *CoRR*, abs/1503.02101, 2015. URL http://arxiv.org/abs/1503.02101.

Golowich, N., Rakhlin, A., and Shamir, O. Size-Independent Sample Complexity of Neural Networks. *arXiv e-prints*, art. arXiv:1712.06541, Dec 2017.

Hardt, M. and Ma, T. Identity matters in deep learning. *CoRR*, abs/1611.04231, 2016.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. How to escape saddle points efficiently. *CoRR*, abs/1703.00887, 2017. URL http://arxiv.org/abs/1703.00887.

Lee, J. D., Simchowitz, M., Jordan, M. I., and Recht, B. Gradient descent only converges to minimizers. In Feldman, V., Rakhlin, A., and Shamir, O. (eds.), *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pp. 1246–1257, Columbia University, New York, New York, USA, 23–26 Jun 2016. PMLR. URL http://proceedings.mlr.press/v49/lee16.html.

Li, Y. and Liang, Y. Learning overparameterized neural networks via stochastic gradient descent on structured data. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 8157–8166. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/8038-learning-overparameterized-neural-networks-v pdf.

Li, Y. and Yuan, Y. Convergence analysis of two-layer neural networks with relu activation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 597–607, USA, 2017. Curran Associates Inc. ISBN 978-1-5108-6096-4. URL http://dl.acm.org/citation.cfm?id=3294771.3294828.

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. Exploring generalization in deep learning. *arXiv:1706.08947*, 2017.

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

Rosset, S., Zhu, J., and Hastie, T. 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]*, pp. 1237–1244, 2003.

Salimans, T. and Kingm, D. P. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *Advances in Neural Information Processing Systems*, 2016.

Shawe-Taylor, J. and Cristianini, N. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521813972.

Sokolic, J., Giryes, R., Sapiro, G., and Rodrigues, M. Robust large margin deep neural networks. *arXiv:1605.08254*, 2017.

Soltanolkotabi, M., Javanmard, A., and Lee, J. D. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 65(2):742–769, Feb 2019. ISSN 0018-9448. doi: 10.1109/TIT.2018.2854560.

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

Tian, Y. 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, pp. 3404–3413. JMLR.org, 2017. URL http://dl.acm.org/citation.cfm?id=3305890.3306033.

Wei, C., Lee, J. D., Liu, Q., and Ma, T. On the Margin Theory of Feedforward Neural Networks. *arXiv e-prints*, art. arXiv:1810.05369, Oct 2018.

Wei, C., Lee, J. D., Liu, Q., and Ma, T. On the margin theory of feedforward neural networks. *CoRR*, abs/1810.05369, 2018.

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

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

Zhong, K., Song, Z., and Dhillon, I. S. Learning non-overlapping convolutional neural networks with multiple kernels. *CoRR*, abs/1711.03440, 2017a. URL http://arxiv.org/abs/1711.03440.

Zhong, K., Song, Z., Jain, P., Bartlett, P. L., and Dhillon, I. S. Recovery guarantees for one-hidden-layer neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 4140–4149. JMLR.org, 2017b. URL http://dl.acm.org/citation.cfm?id=3305890.3306109.

Zou, D., Cao, Y., Zhou, D., and Gu, Q. Stochastic gradient descent optimizes over-parameterized deep relu networks. *CoRR*, abs/1811.08888, 2018. URL http://arxiv.org/abs/1811.08888.