Convex Optimization for Shallow Neural Networks

Tolga Ergen and Mert Pilanci
Department of Electrical Engineering, Stanford University
Email: {ergen, pilanci}@stanford.edu

***Abstract*— We consider non-convex training of shallow neural networks and introduce a convex relaxation approach with theoretical guarantees. For the single neuron case, we prove that the relaxation preserves the location of the global minimum under a planted model assumption. Therefore, a globally optimal solution can be efficiently found via a gradient method. We show that gradient descent applied on the relaxation always outperforms gradient descent on the original non-convex loss with no additional computational cost. We then characterize this relaxation as a regularizer and further introduce extensions to multineuron single hidden layer networks.**

I. Introduction

Deep neural networks have attracted significant attention due to their success in various applications, e.g., computer vision [1] and natural language processing [2]. Despite their highly non-convex and nonlinear structure, training of these networks is adequately performed by simple first-order gradient based optimization algorithms such as Gradient Descent (GD). However, it is still theoretically elusive that how such algorithms can successfully train deep networks and obtain solutions that generalize.

In order to resolve this issue, theoretical aspects of training neural networks with GD have been extensively studied recently. Since shallow networks can approximate any function with a sufficient number of neurons [3] and have a comparatively simpler structure to analyze, most of the theoretical results in the current literature are based on shallow networks. Among these, [4]–[7] studied the recovery of a planted model parameter for one hidden layer networks trained with GD, where they used expected risk minimization. In particular, [4] proved that for one neuron networks with the Rectified Linear Unit (ReLU) activation, GD with random initialization can recover the planted model parameter. In a similar work, [5] showed that GD can recover the planted parameters of an NN with one hidden layer. Then, [6] proved that Stochastic GD (SGD) with random initialization can learn one layer NNs in polynomial time. Later on, [7] extended the previous work to two layer training, where the input is assumed to be Gaussian. However, all of these studies are based on expected risk minimization, which is not practical since we have only training samples in practice [8]. Additionally, most of the studies either work under the Gaussian input assumption or provide restricted theoretical results without this assumption, e.g., [9].

Unlike the above studies, [10]–[12] focused on directly minimizing the generic ℓ_2 norm loss function, which is also known as empirical risk. However, [10]–[12] provided only a locally linear convergence rate for smooth activation functions, where they also require a proper initialization. Furthermore, these results cannot be applied to most applications since some commonly used activation functions, e.g., ReLU, are non-smooth.

Another line of research [13]–[15] studied overparameterized neural networks. In particular, [13] proved the convergence rate of SGD for a two layer network with leaky ReLU activations in an overparameterized scenario. However, they also assumed that the input data is linearly separable, which is not the case in general. [14] further relaxed this assumption to the case where the data is well clustered. However, in their results, the amount of overparameterization depends on the desired error value. Later on, [15] remedied this issue by providing a result for GD, where the amount of overparameterization is independent of the desired error value. However, these studies assume that the number of neurons is extremely large, which is an unrealistic assumption.

In this paper, we provide a theoretical analysis for the globally optimal training of a wide range of NN architectures without requiring strong assumptions. Our main contributions can be summarized as follows: 1) We study the training of shallow neural networks using empirical risk minimization. Thus, unlike [4]–[7], we present a theoretical analysis which is more practical; 2) We analyze shallow networks and illustrate why GD might fail to achieve the global optimum of the training cost. Based on our analysis, we derive a convex relaxation for the original loss function for a single neuron. We then prove that the introduced relaxation preserves the location of the global minimum under a planted model assumption; 3) We extend these observations to the multineuron case by characterizing the relaxation as a regularization term. We then prove that the regularized GD achieves the optimal training performance for the original non-convex loss.

Notation: We denote the matrices and vectors (or scalars) as uppercase and lowercase letters, respectively. We also use $\|\cdot\|_2$ to denote the Euclidean norm. To denote a vector or matrix of zeros or ones, we use 0 or 1, respectively, where the sizes are understood from the context. Additionally, I_s represents the identity matrix of the size s.

II. Main Section

Here, we consider the following minimization problem

$$\hat{x} = \operatorname{argmin} f(x),$$

where $\hat{x} \in \mathbb{R}^d$ represents the desired parameter vector and $f(x)$ is the objective function. Throughout the paper, we

assume that $f(\cdot)$ is the ℓ_2 norm loss function.

A. Single Neuron Training

In this section, we aim to solve the following problem

$$\arg\min_x \frac{1}{2} \underbrace{\|g(Ax) - y\|_2^2}_{f(x)}, \tag{1}$$

where $g(\cdot)$ is a nonlinear activation function, $A \in \mathbb{R}^{n\times d}$ is the data matrix, $x \in \mathbb{R}^d$ is the parameter matrix, and $y \in \mathbb{R}^n$ is the observation matrix. We let $\{a_i\}_{i=1}^n$ denote the rows of the data matrix A, which are d dimensional feature vectors. We can solve (1) using the GD algorithm as follows

$$x_{t+1} = x_t - \mu A^T D_t(g(Ax_t) - y), \tag{2}$$

where μ is the learning rate, the subscript t is the iteration index, D_t is a diagonal matrix and its i^{th} diagonal entry is computed as 1 if $a_i^T x_t \geq 0$, 0 otherwise.

For ReLU function, i.e., $g(x) = \max\{0, x\}$, when we express $f(x)$ in a different form as

$$f(x) = \|g(Ax)\|_2^2 - 2y^T g(Ax) + \|y\|_2^2,$$

we observe that our problem is non-convex due to the second term above. In order to make the problem convex (see Proposition 1), we relax $f(x)$ as follows

$$f_r(x) = \|g(Ax)\|_2^2 - 2y^T Ax + \|y\|_2^2. \tag{3}$$

Here, we consider the following planted model

$$y = g(Ax^*), \tag{4}$$

where x^* is the planted parameter vector.

Proposition 1. *Provided that* (4) *hold, the relaxed version of the objective function $f(x)$ is convex.*

Proof. Here, we directly use the convexity definition to prove the convexity of $f_r(x)$ as in the following. Let $u, v \in \mathbb{R}^d$, then $\forall \alpha \in [0, 1]$, we have

$$\begin{aligned} f_r(\alpha u + (1-\alpha))v) &= \|g(A(\alpha u + (1-\alpha)v))\|_2^2 \\ &\quad - 2y^T(A(\alpha u + (1-\alpha)v)) + \|y\|_2^2 \\ &\leq \|g(\alpha Au)\|_2^2 + \|g((1-\alpha)Av)\|_2^2 \\ &\quad - 2y^T\big(\alpha(Au) + (1-\alpha)(Av)\big) \\ &\quad + \|y\|_2^2 \\ &\leq \alpha f_r(u) + (1-\alpha) f_r(v), \end{aligned}$$

which concludes the proof. □

Thus, we can find the global minimum of $f_r(x)$ using the GD algorithm. Based on the definition of the new objective function $f_r(x)$, we modify the GD update in (2) as

$$x_{t+1} = x_t - \mu A^T(g(Ax_t) - y) \tag{5}$$

Theorem 1. *Given the model* (4)*, let $\tilde{x}$ be the global minimum of $f_r(x)$. Then, $\tilde{x}$ is a global minimizer of $f(x)$, i.e., $f(x^*) = f(\tilde{x})$, and* $\min f(x) = \min f_r(x)$.

Proof. We first note that $f_r(x)$ is an upper-bound for $f(x)$, i.e., $f_r(x) \geq f(x), \forall x \in \mathbb{R}^d$, due to its definition in (3). Since x^* is the minimum of $f(x)$, we have

$$f_r(\tilde{x}) \geq f(\tilde{x}) \geq f(x^*). \tag{6}$$

Additionally, we have

$$\begin{aligned} f_r(x^*) &= \|g(Ax^*)\|_2^2 - 2y^T g(Ax^*) + \|y\|_2^2 \\ &= \|g(Ax^*)\|_2^2 - 2y^T(Ax^*) + \|y\|_2^2 \\ &= f(x^*), \end{aligned}$$

where the second equality follows from our planted model, i.e., $y = g(Ax^*)$, and the properties of the max function, i.e., $max\{0, u\}u = max\{0, u\}^2$, $\forall u \in \mathbb{R}$. With this observation, we obtain the following inequality

$$f_r(\tilde{x}) \leq f_r(x^*) = f(x^*). \tag{7}$$

By (6) and (7), we have $f_r(\tilde{x}) = f(x^*)$, which completes the proof. □

B. Multineuron Training

In this section, instead of (4), we assume that the observations come from the following model

$$y_i = \sum_{j=1}^m g(a_i^T x_j^*). \tag{8}$$

For (8), we aim to minimize the following function

$$\arg\min_X \frac{1}{2} \underbrace{\Big\| \sum_{j=1}^m g(Ax_j) - y \Big\|_2^2}_{f_m(X)}, \tag{9}$$

where we denote the observation vector as y and the j^{th} column of the parameter matrix $X \in \mathbb{R}^{d\times m}$ as x_j.

We first note that (9) has several global minimum points. As an example, we can change the order of the neurons, however, y will still be the same. Thus, we cannot make it convex. However, we can still improve the performance. For $n > d$, since we can obtain more equations than the number of parameters, the true parameter can be achieved even by the conventional GD algorithm [12]. However, when $n \leq d$, GD might fail to achieve the global minimum. Thus, we examine the $n \leq d$ case and propose an algorithm that achieves the global minimum at any stationary point.

Let us assume that $g(\cdot)$ is the ReLU function in order to illustrate why GD might fail to achieve the global minimum when $n \leq d$. With the ReLU assumption, we know that $\nabla_{x_j} f_m(X) = 2A^T D_j r$, where r is the residual, i.e.,

$$r = \sum_{j=1}^m g(Ax_j) - y. \tag{10}$$

We also know that at a local or global minimum, i.e., a stationary point, all the parameter vectors satisfy

$$\nabla_{x_j} f_m(X) = 0, \forall j \in \{1, 2, \ldots, m\}.$$

If we further assume that A is full rank, which holds with high probability for random matrices [16], we obtain the following gradient for each neuron j

$$\nabla_{x_j} f_m(X) = 2A^T D_j r = 0 \tag{11}$$

if and only if

$$D_j r = 0,\ \forall j \in \{1, 2, \ldots, m\}. \tag{12}$$

The above equalities imply that the conventional GD approach only guarantees that an entry of the residual surely vanishes if the corresponding entry of the diagonal matrix is active for at least one neuron. Mathematically, this claim corresponds to the following

$$\sum_{j=1}^{m} D_j r = 0,$$

which does not guarantee that the residual indexes that correspond to a 0 element of all D_j's vanish by using the conventional GD algorithm. Note that these indexes can be represented as $\prod_{j=1}^{m} D_j^c$, where $D_j^c = I_n - D_j$.

In order to enforce GD to minimize these indexes as well, we introduce a new term to the gradient as follows

$$\nabla_{x_j} f_{mp}(X) = 2A^T D_j r + 2A^T \prod_{j=1}^{m} D_j^c r. \tag{13}$$

With this modification, we can guarantee that each index of the residual has a positive multiplicative factor so that $\nabla_{x_j} f_{mp}(X) = 0,\ \forall j$ implies that $r = 0$. In other words, any stationary point of the new objective function $f_{mp}(X)$ is a global minimum.

In the following, we propose a function that satisfies (13) and prove that global minimums of this function are the same with the global minimums of $f_m(X)$.

Proposition 2. *The following selection for $f_{mp}(X)$ yields the desired gradient* (13) *when the entries of AX is nonzero.*

$$f_{mp}(X) = \Big\| \sum_{j=1}^{m} g(Ax_j) - y \Big\|_2^2 - 2y^T \prod_{j=1}^{m} D_j^c A \sum_{j=1}^{m} x_j. \tag{14}$$

Proof. When we take the gradient of (14) with respect to x_j, we obtain

$$\begin{aligned}\nabla_{x_j} f_{mp}(X) &= 2A^T D_j r - 2A^T \prod_{j=1}^{m} D_j^c y + \gamma(X, A) \\ &= 2A^T D_j r + 2A^T \prod_{j=1}^{m} D_j^c r,\end{aligned}$$

where $\gamma(X, A)$ represents the remaining terms of the gradient due to the chain rule. However, since we assume that AX has no zero entries, $\gamma(X, A) = 0$, which thus vanishes. For the second equality, we first provide an alternative definition of the ReLU function as $g(Ax_j) = D_j Ax$. Then, due to $D_j^c D_j = 0$, our model in (8), and (10), we have

$$\prod_{j=1}^{m} D_j^c r = -\prod_{j=1}^{m} D_j^c y,$$

which yields the result. □

Theorem 2. *For* (8)*, let $\tilde{X}$ be a global minimum of $f_{mp}(X)$. Then, $\tilde{X}$ is a global minimizer of $f_m(X)$, i.e., $f_m(X^*) = f_m(\tilde{X})$, and $\min f_m(X) = \min f_{mp}(X)$.*

Proof. We first note that $f_{mp}(X)$ is an upper-bound for $f_m(X)$, i.e., $f_{mp}(X) \geq f_m(X), \forall X \in \mathbb{R}^{d \times m}$, due to (14). Since X^* is the minimum of $f_m(X)$, we have

$$f_{mp}(\tilde{X}) \geq f_m(\tilde{X}) \geq f_m(X^*). \tag{15}$$

Additionally, we have

$$\begin{aligned}f_{mp}(X^*) &= \Big\| \sum_{j=1}^{m} g(Ax_j^*) - y \Big\|_2^2 - 2y^T \prod_{j=1}^{m} D_j^{*^c} A \sum_{j=1}^{m} x_j^* \\ &= \Big\| \sum_{j=1}^{m} g(Ax_j^*) - y \Big\|_2^2 \\ &= f_m(X^*),\end{aligned}$$

where the second equality follows from our planted model, i.e., $y = \sum_{j=1}^{m} g(Ax_j^*)$, and $D_j^{*^c} D_j^* = 0,\ \forall j$. With this observation, we obtain the following inequality

$$f_{mp}(\tilde{X}) \leq f_{mp}(X^*) = f_m(X^*). \tag{16}$$

By (15) and (16), we have $f_{mp}(\tilde{X}) = f_m(X^*)$, which completes the proof. □

Corollary 1. *As a result of Theorem 2 and the definition in* (13)*, whenever the following update converges, it achieves the global minimum of $f_m(X)$*

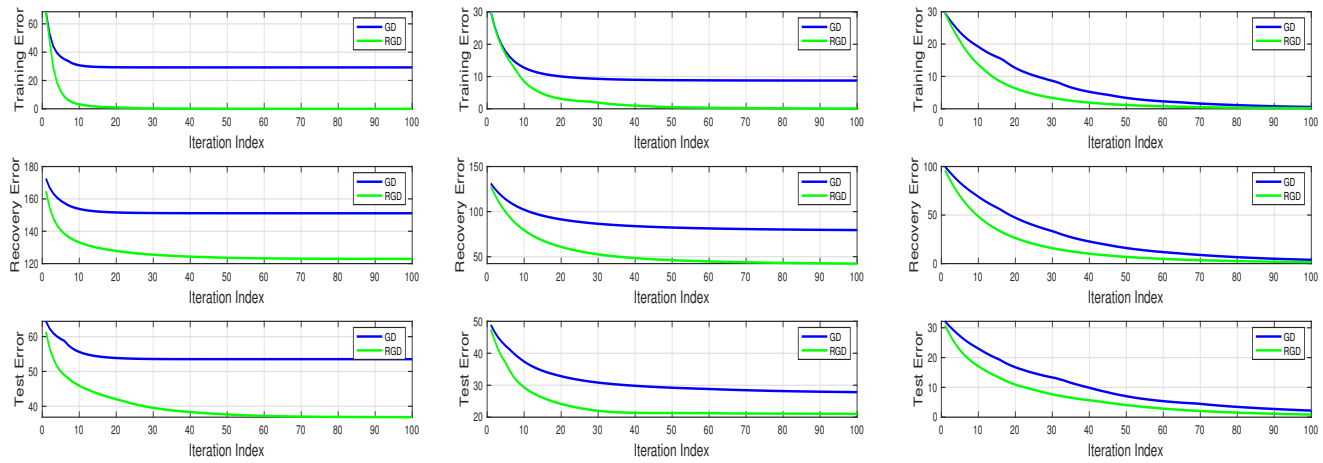
$$x_{t+1,j} = x_{t,j} - \mu \nabla_{x_j} f_{mp}(X_t) \tag{17}$$

for each neuron j.

III. Numerical Experiments

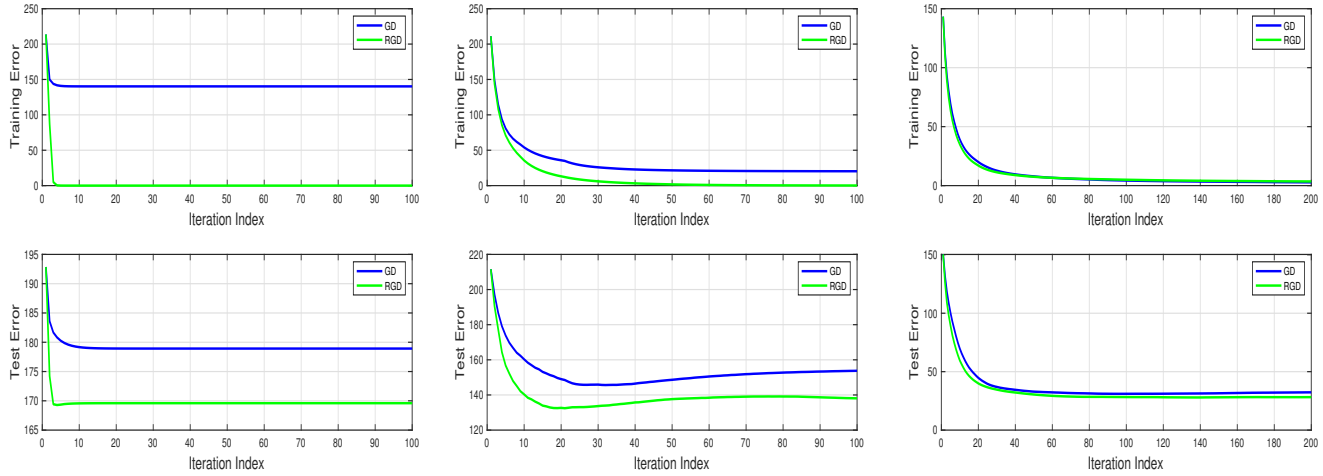
In this section, we use synthetic data that obeys the models in (4) and (8) in order to verify our theoretical analysis. For the experiments in this section, we obtain the data matrices using a multivariate Gaussian distribution with zero mean and identity covariance matrix, i.e., $a_i \sim N(0, I_d)$ for $i = 1, 2, \ldots, m$, unless otherwise stated. In addition to this, we randomly initialized the parameter vector, i.e., $x_0 \sim N(0, I_d)$, and use the ReLU as activation. Throughout the section, we denote our approach and conventional GD as Regularized GD (RGD) and GD, respectively.

We first consider the single neuron training performance of RGD and GD, where we particularly focus on training, test, and recovery errors. Note that in this context, recovery error is computed as the Euclidean distance between the planted model parameter and the estimated parameter vector. Here, we choose the learning rate as $\mu = 0.1$ for both algorithms.



(a) $m = 1, n = 50,$ and $d = 100$ **(b)** $m = 1, n = 50,$ and $d = 50$ **(c)** $m = 1, n = 200,$ and $d = 50$

Fig. 1: Comparison of GD and RGD in terms of the training, test and true parameter recovery errors when $m = 1$.



(a) $m = 3, n = 10,$ and $d = 100$ **(b)** $m = 3, n = 100,$ and $d = 100$ **(c)** $m = 3, n = 200,$ and $d = 50$

Fig. 2: Comparison of GD and RGD in terms of the training and test errors when $m = 3$.

Additionally, we use 100 samples for the test data. In Fig. 1, we illustrate three scenarios, i.e., $n < d$, $n = d$, and $n > d$. In Fig. 1a and 1b, RGD achieves zero training error when $n \leq d$, which matches with our analysis. Moreover, RGD provides smaller error for the recovery of the true parameter and the test case. When $n > d$, since both algorithms have enough nonzero observations, they achieve zero error for each case in Fig. 1c. However, our regularization enables RGD to converge at a faster rate.

We also consider the multineuron training case. Here, since the permutations of neurons do not change the value of the objective function, there are multiple global minimums. Hence, we do not consider the recovery error unlike the single neuron training case. For both algorithms, we choose $\mu = 0.08$ and use 3 neurons, i.e., $m = 3$. Note that in (13), our contribution depends on a multiplicative term, where we multiply m diagonal matrices that has 0's and 1's at their diagonal. Therefore, as m increases, the effect of our contribution will get smaller. We also observe this effect in Fig. 2, where as n increases, the performance gap between RGD and GD tends to vanish at a faster rate. Other than this, we again observe that RGD achieves zero training error and lower test error for each case while GD only provides a comparable performance when $n > d$.

IV. Conclusion

In this paper, we studied globally optimal training of shallow networks, where we used the ℓ_2 norm loss function. In order to achieve the globally optimal training performance, we first introduced a convex relaxation for the original non-convex loss function of a single neuron network. We then proved that this relaxation preserves the locations of the

global minimum under a planted model assumption. We then characterize this relaxation as a regularization term for the gradient of the original loss function in the case of multi-neuron. We also proved that this regularization preserves the global minimums of the original function. We then verified our theoretical results using numerical experiments. There are several open directions for future work. Extending our framework for more general activation functions beyond ReLU and deeper neural networks is currently an open problem. We also refer the reader to [17] for a block-wise approach to multilayer networks. Furthermore, random projection and sketching methods recently developed for convex optimization problems [18]–[21] can also be employed on convex neural network relaxations for faster training.

ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation under grant IIS-1838179.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, Nov 2012.

[3] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251 – 257, 1991. [Online]. Available: http://www.sciencedirect.com/science/article/pii/089360809190009T

[4] Y. Tian, "An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis," *arXiv preprint arXiv:1703.00560*, 2017.

[5] A. Brutzkus and A. Globerson, "Globally optimal gradient descent for a convnet with gaussian inputs," *arXiv preprint arXiv:1702.07966*, 2017.

[6] S. S. Du, J. D. Lee, and Y. Tian, "When is a convolutional filter easy to learn?" *arXiv preprint arXiv:1709.06129*, 2017.

[7] S. Du, J. Lee, Y. Tian, A. Singh, and B. Poczos, "Gradient descent learns one-hidden-layer CNN: Dont be afraid of spurious local minima," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. Stockholmsmssan, Stockholm Sweden: PMLR, 10–15 Jul 2018, pp. 1339–1348. [Online]. Available: http://proceedings.mlr.press/v80/du18b.html

[8] X. Zhang, Y. Yu, L. Wang, and Q. Gu, "Learning one-hidden-layer relu networks via gradient descent," *arXiv preprint arXiv:1806.07808*, 2018.

[9] B. Bartan and M. Pilanci, "Convex relaxations of convolutional neural nets," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 4928–4932.

[10] 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*, 2018.

[11] H. Fu, Y. Chi, and Y. Liang, "Local geometry of one-hidden-layer neural networks for logistic regression," *arXiv preprint arXiv:1802.06463*, 2018.

[12] K. Zhong, Z. Song, P. Jain, P. L. Bartlett, and I. S. Dhillon, "Recovery guarantees for one-hidden-layer neural networks," *arXiv preprint arXiv:1706.03175*, 2017.

[13] A. Brutzkus, A. Globerson, E. Malach, and S. Shalev-Shwartz, "SGD learns over-parameterized networks that provably generalize on linearly separable data," *CoRR*, vol. abs/1710.10174, 2017. [Online]. Available: http://arxiv.org/abs/1710.10174

[14] Y. Li and Y. Liang, "Learning overparameterized neural networks via stochastic gradient descent on structured data," *CoRR*, vol. abs/1808.01204, 2018. [Online]. Available: http://arxiv.org/abs/1808.01204

[15] S. S. Du, X. Zhai, B. Póczos, and A. Singh, "Gradient descent provably optimizes over-parameterized neural networks," *CoRR*, vol. abs/1810.02054, 2018. [Online]. Available: http://arxiv.org/abs/1810.02054

[16] X. Feng and Z. Zhang, "The rank of a random matrix," *Applied Mathematics and Computation*, vol. 185, no. 1, pp. 689 – 694, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0096300306009040

[17] F. Gu, A. Askari, and L. E. Ghaoui, "Fenchel lifted networks: A lagrange relaxation of neural network training," *arXiv preprint arXiv:1811.08039*, 2018.

[18] M. Pilanci and M. J. Wainwright, "Randomized sketches of convex programs with sharp guarantees," *IEEE Trans. Info. Theory*, vol. 9, no. 61, pp. 5096–5115, September 2015.

[19] ——, "Iterative Hessian sketch: Fast and accurate solution approximation for constrained least-squares," UC Berkeley, Tech. Rep., 2014, full length version at arXiv:1411.0347.

[20] ——, "Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence," *SIAM Journal on Optimization*, vol. 27, no. 1, pp. 205–245, 2017.

[21] Y. Yang, M. Pilanci, M. J. Wainwright, *et al.*, "Randomized sketches for kernels: Fast and optimal nonparametric regression," *The Annals of Statistics*, vol. 45, no. 3, pp. 991–1023, 2017.