Towards Provable Learning of Polynomial Neural Networks Using Low-Rank Matrix Estimation

Mohammadreza Soltani
Iowa State University
msoltani@iastate.edu

Chinmay Hegde
Iowa State University
chinmay@iastate.edu

Abstract

We study the problem of (provably) learning the weights of a two-layer neural network with quadratic activations. In particular, we focus on the under-parametrized regime where the number of neurons in the hidden layer is (much) smaller than the dimension of the input. Our approach uses a lifting trick, which enables us to borrow algorithmic ideas from low-rank matrix estimation. In this context, we propose a novel, non-convex training algorithm which do not need any extra tuning parameters other than the number of hidden neurons. We support our algorithm with rigorous theoretical analysis, and show that the proposed algorithm enjoy linear convergence, fast running time per iteration, and near-optimal sample complexity.

1 Introduction

In this paper, we (provably) resolve several algorithmic challenges that arise in the context of a special class of (shallow) neural networks by making connections to the better-studied problem of *low-rank matrix estimation*. Our hope is that a rigorous understanding of the fundamental limits of training shallow networks can be used as building blocks to obtain theoretical insights for more complex networks.

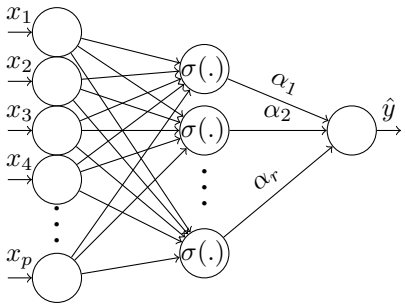

Figure 1: *Two-layer polynomial neural network.*

Consider a shallow (two-layer) neural network architecture, as illustrated in Figure 1. This network comprises p input nodes, a single hidden layer with r neurons with activation function $\sigma(z)$, first layer weights $\{w_j\}_{j=1}^r \subset \mathbb{R}^p$, and an output layer comprising of a single node and weights $\{\alpha_j\}_{j=1}^r \subset \mathbb{R}$. If $\sigma(z) = z^2$, then the above network is called a *polynomial neural network* [1]. More precisely, the input-output relationship between an input, $x \in \mathbb{R}^p$, and the corresponding output, $y \in \mathbb{R}$, is given by:

$$\hat{y} = \sum_{j=1}^{r} \alpha_j \sigma(w_j^T x) = \sum_{j=1}^{r} \alpha_j \langle w_j, x \rangle^2.$$

Workshop On Deep Learning: Bridging Theory and Practice (DLP), NIPS 2017, Long Beach, CA, USA.

Here, our focus is in the so-called "under-parameterized" regime where $r \ll p$. Our goal is to learn this network, given a set of training input-output pairs $\{(x_i, y_i)\}_{i=1}^m$. We do so by finding a set of weights $\{\alpha_j, w_j\}_{j=1}^r$ that minimize the following *empirical risk*:

$$\min_{W \in \mathbb{R}^{r \times p}, \alpha \in \mathbb{R}^r} \quad F(W, \alpha) = \frac{1}{2m} \sum_{i=1}^{m} \left(y_i - \hat{y}_i\right)^2, \tag{1}$$

where the rows of W and the entries of α indicate the first-and second-layer weights, respectively. Numerous recent papers have explored (provable) algorithms to learn the weights of such a network under distributional assumptions on the input data [1, 2, 3, 4, 5, 6, 7][1]. Clearly, the empirical risk defined in (1) is extremely non-convex (involving fourth-powers of the entries of w_j, coupled with the squares of α_j). However, this can be circumvented using a clever *lifting* trick: if we define the matrix variable $L_* = \sum_{j=1}^r \alpha_j w_j w_j^T$, then the input-output relationship becomes:

$$\hat{y}_i = x_i^T L_* x_i = \langle x_i x_i^T, L_* \rangle, \tag{2}$$

where $x_i \in \mathbb{R}^p$ denotes the i^{th} training sample. Moreover, the variable L_* is a rank-r matrix of size $p \times p$. Therefore, (1) can be viewed as an instance of learning a fixed (but unknown) rank-r symmetric matrix $L_* \in \mathbb{R}^{p \times p}$ with $r \ll p$, from a small number of *rank-one* linear observations given by $A_i = x_i x_i^T$. While still non-convex, low-rank matrix estimation problems such as (2) are much better understood.

In this paper, we make concrete algorithmic progress on solving low-rank matrix estimation problems of the form (2). In the context of learning polynomial neural networks, once we have estimated a rank-r symmetric matrix L_*, we can always produce weights $\{\alpha_j, w_j\}$ by an eigendecomposition of L_*. In general, a range of algorithms for solving (2) (or variants thereof) exist in the literature, and can be broadly classified into two categories: (i) *convex* approaches, all of which involve enforcing the rank-r assumption in terms of a convex penalty term, such as the nuclear norm [8, 9, 10, 11, 12]; (ii) *nonconvex* approaches based on either alternating minimization [13, 2] or greedy approximation [1, 14]. Both types of approaches suffer from severe computational difficulties, particularly when the data dimension p is large. Even the most computationally efficient convex approaches require multiple invocations of singular value decomposition (SVD) of a (potentially) large $p \times p$ matrix, which can incur *cubic* ($O(p^3)$) running time. Moreover, even the best available non-convex approaches require a very accurate initialization, and also require that the underlying matrix L_* is well-conditioned; if this is not the case, the running time of all available methods again inflates to $O(p^3)$, or worse.

In this paper, we take a different approach, and show how to leverage recent results in low-rank approximation to our advantage [15, 16]. In addition, by using the idea of fresh sampling (which only increases the number of iterations logarithmically), we show that while sensing operator does not satisfy RIP condition, it satisfies similar property by careful unbiasing argument. Based on these ideas, we propose an algorithm which is also non-convex; however, unlike all earlier works, it does not require any full SVD calculations. Specifically, we demonstrate that a careful concatenation of *randomized, approximate* SVD methods, coupled with appropriately defined gradient steps, leads to efficient and accurate matrix estimation.

To our knowledge, this work constitutes the first *nearly-linear* time method for low-rank matrix recovery from rank-one observations. Consequently, in the context of learning two-layer polynomial networks, our method is the first to exhibit *nearly-linear* running time, is *nearly sample-optimal* for fixed target rank r, and is *unconditional* (i.e., it makes no assumptions on the condition number of L_* or the weight matrix W).

2 Main Results

Throughout this paper, $\|\cdot\|_F$ and $\|\cdot\|_2$ denote the matrix Frobenius and spectral norm, respectively. The phrase "with high probability" indicates an event whose failure rate is exponentially small. We assume that the training data samples (x, y) obey a generative model (2) written as:

$$y = \sum_{j=1}^{r} \alpha_j^* \sigma(\langle w_j^*, x \rangle) = x^T L_* x + e \tag{3}$$

[1] While quadratic activation functions are rarely used in practice, stacking multiple such two-layer blocks can be used to simulate networks with higher-order polynomial and sigmoidal activations [1].

Algorithm 1 AP-ROM

Inputs: y, number of iterations K, independent data samples $\{x_1^t, x_2^t \ldots, x_m^t\}$ for $t = 1, \ldots, K$, rank r
Outputs: Estimates $\widehat{L}$
Initialization: $L_0 \leftarrow 0$, $t \leftarrow 0$
Calculate: $\bar{y} = \frac{1}{m}\sum_{i=1}^{m} y_i$
while $t \leq K$ **do**
$L_{t+1} = \mathcal{T}\left(L_t - \mathcal{H}\left(\frac{1}{2m}\sum_{i=1}^{2m}\left((x_i^t)^T L_t x_i^t - y_i\right) x_i^t (x_i^t)^T - (\frac{1}{2m}\mathbf{1}^T\mathcal{A}(L_t) - \frac{1}{2}\bar{y})I\right)\right)$
$t \leftarrow t + 1$
end while
Return: $\widehat{L} = L_K$

where $L_* \in \mathbb{R}^{p\times p}$ is the "ground-truth" matrix (with rank equal to r). Define $\mathcal{A} : \mathbb{R}^{p\times p} \to \mathbb{R}^m$ such that: $\mathcal{A}(L_*) = [x_1^T L_* x_1, x_2^T L_* x_2, \ldots, x_m^T L_* x_m]^T$, and each $x_i \overset{i.i.d}{\sim} \mathcal{N}(0, I)$ is a normal random vector in $\mathbb{R}^p$ for $i = 1, \ldots, m$. The adjoint operator of $\mathcal{A}$ is defined as $\mathcal{A}^*(y) = \sum_{i=1}^m y_i x_i x_i^T$. Here, $e \in \mathbb{R}^m$ denotes an additive noise term; throughout the paper (for the purpose of analysis) we assume that e is zero-mean, subgaussian with i.i.d entries, and independent of x_i's. The goal is to learn the rank-r matrix parameter L^* from as few samples as possible. In our analysis, we require the operators $\mathcal{A}$ and $\mathcal{A}^*$ to satisfy the following regularity condition with respect to the set of low-rank matrices. We call this the *Conditional Unbiased Restricted Isometry Property*, abbreviated as *CU-RIP*(ρ):

Definition 1. *Consider fixed rank-r matrices L_1 and L_2. Then, $\mathcal{A}$ is said to satisfy CU-RIP(ρ) if there exists $0 < \rho < 1$ such that*

$$\left\| L_1 - L_2 - \frac{1}{2m}\mathcal{A}^*\mathcal{A}(L_1 - L_2) - \frac{1}{2m}\mathbf{1}^T\left(\mathcal{A}(L_1) - \mathcal{A}(L_2)\right) I \right\|_2 \leq \rho \left\| L_1 - L_2 \right\|_2.$$

where $\mathbf{1}$ denotes a vector in $\mathbb{R}^m$ with entries all equal to 1. Let $\mathbb{U}_r$ denote the set of all rank-r matrix subspaces. We use the idea of *head* and *tail* approximate projections with respect to $\mathbb{U}_r$ first proposed in [17], and instantiated in the context of low-rank approximation in [16].

Definition 2 (Approximate tail projection)**.** *$\mathcal{T} : \mathbb{R}^{p\times p} \to \mathbb{U}_r$ is a ε-approximate tail projection algorithm if for all $L \in \mathbb{R}^{p\times p}$, $\mathcal{T}$ returns a subspace $W = \mathcal{T}(L)$ that satisfies: $\|L - \mathcal{P}_W L\|_F \leq (1+\varepsilon)\|L - L_r\|_F$, where L_r is the optimal rank-r approximation of L.*

Definition 3 (Approximate head projection)**.** *$\mathcal{H} : \mathbb{R}^{p\times p} \to \mathbb{U}_r$ is a ε-approximate head projection if for all $L \in \mathbb{R}^{p\times p}$, the returned subspace $V = \mathcal{H}(L)$ satisfies: $\|\mathcal{P}_V L\|_F \geq (1-\varepsilon)\|L_r\|_F$, where L_r is the optimal rank-r approximation of L.*

In general, we propose two methods to estimate L_* given knowledge of $\{x_i, y_i\}_{i=1}^m$. However, our first method is somewhat computationally inefficient, but achieves very good sample complexity and serves to illustrate the overall algorithmic approach. Due to lack of space, we only present our second algorithm, improving the overal running time. For the full discussion about all the algorithms, prior art, proof of all the theoretical results, and the experimental results, please see [18].

Now, consider the non-convex, constrained risk minimization problem:

$$\begin{aligned} \min_{L\in\mathbb{R}^{p\times p}} \quad & F(L) = \frac{1}{2m}\sum_{i=1}^{m}\left(y_i - x_i^T L x_i\right)^2 \\ \text{s.t.} \quad & \text{rank}(L) \leq r. \end{aligned} \tag{4}$$

To solve this problem, we propose our second algorithm that we call *Approximate Projection for Rank One Matrix* recovery, or *AP-ROM*. We display the pseudocode of AP-ROM in Algorithm 1.

In algorithm 1, the specific choice of approximate SVD algorithms that simulate the operators $\mathcal{T}(.)$ and $\mathcal{H}(.)$ is flexible. We note that tail-approximate projections have been widely studied in the numerical linear algebra literature [19, 20, 21]; however, head-approximate projection methods are less well-known. In our method, we use the randomized Block Krylov SVD (BK-SVD) method proposed by [15], which has been shown to satisfy both types of approximation guarantees [16]. One can alternatively use LazySVD, recently proposed by [22], which also satisfies both guarantees. The nice feature of these methods is that their running time is ***independent of the spectral gap*** of the

matrix. We leverage this property to show asymptotic improvements over other fast SVD methods (such as the power method).

We briefly discuss the BK-SVD algorithm. In particular, BK-SVD takes an input matrix with size $p \times p$ with rank r and returns a r-dimensional subspace which approximates the top right r singular vectors of the input. Mathematically, if $A \in \mathbb{R}^{p\times p}$ is the input, A_r is the best rank-r approximation to it, and Z is a basis matrix that spans the subspace returned by BK-SVD, then the projection of A into Z, $B = ZZ^TA$ satisfies the following relations:

$$\|A - B\|_F \leq (1+\varepsilon)\|A - A_r\|_F, \quad |u_i^T AA^T u_i - z_i AA^T z_i| \leq \varepsilon\sigma_{r+1}^2,$$

where $\varepsilon > 0$ is defined as the tail and head projection approximate constant, and u_i denotes the i^{th} right eigenvector of A. In Appendix-B of [16], it has been shown that the per-vector guarantee (the left property in above) can be used to prove the approximate head projection property, i.e., $\|B\|_F \geq (1-\varepsilon)\|A_r\|_F$. We now establish that AP-ROM exhibits linear convergence.

Theorem 4 (Convergence of AP-ROM)**.** *Consider the sequence of iterates (L_t) obtained in AP-ROM. Assume that in each iteration t, $\mathcal{A}$ satisfies CU-RIP(ρ'), then AP-ROM outputs a sequence of estimates L_t such that:*

$$\|L_{t+1} - L_*\|_F \leq q_1'\|L_t - L_*\|_F + q_2'\left(|\mathbf{1}^T e| + \left\|\mathcal{A}^* e\right\|_2\right), \tag{5}$$

where $q_1' = (2+\varepsilon)(\rho' + \sqrt{1-\phi^2})$, $q_2' = \frac{\sqrt{r}}{2m}\left(2 - \varepsilon + \frac{\phi(2-\varepsilon)(2+\varepsilon)}{\sqrt{1-\phi^2}}\right)$, and $\phi = (1-\varepsilon)(1-\rho') - \rho'$.

In the full version of this work, [18], we showed that CU-RIP is satisfied in each iteration of AP-ROM with probability at least $1-\xi$, provided that $m = \mathcal{O}\left(\frac{1}{\delta^2} pr^3 \log^3 p \log(\frac{p}{\xi})\right)$ for some $\delta > 0$. In addition, we gave an upper bound in the statistical term in (5) involving noise vector, e. Overall, we have the following result:

Corollary 5. *The output of AP-ROM satisfies the following after K iterations with high probability:*

$$\|L_K - L_*\|_F \leq (q_1')^K \|L_*\|_F + \frac{C_\tau'' q_2'}{1 - q_1'}\sqrt{\frac{pr\log^3 p}{m}}. \tag{6}$$

where q_1' and q_2' have been defined in Theorem 4.

Hence, under the assumptions in Theorem 4, in order to achieve ϵ-accuracy in the estimation of L_* in terms of Frobenius norm, AP-ROM requires $K = \mathcal{O}(\log(\frac{\|L_*\|_2}{\epsilon}))$ iterations. From Theorem 5 and Corollary 5, we observe that the sample-complexity of AP-ROM (i.e., the number of samples m to achieve a given accuracy) scales as $m = \mathcal{O}\left(pr^3 \log^4 p \log(\frac{1}{\epsilon})\right)$.

The above analysis of AP-ROM shows that instead of using exact rank-r projections, one can use instead tail and head approximate projection which is implemented by the BK-SVD method of [15]. The running time for this method is given by $\widetilde{\mathcal{O}}(p^2 r)$ if $r \ll p$. While the running time of the projection step is gap-independent, the calculation of the *gradient* (i.e., the input to the head projection method $\mathcal{H}$) is itself the major bottleneck. In essence, this is related to the calculation of the adjoint operator, $\mathcal{A}^*(d) = \sum_{i=1}^m d^{(i)} x_i x_i^T$, which requires $\mathcal{O}(p^2)$ operations for each sample. Coupled with the sample-complexity of $m = \Omega(pr^3)$, this means that the running time per-iteration scales as $\Omega(p^3 r^3)$, which overshadows any gains achieved during the projection step (please see [18] for more discussion). To address this challenge, we propose a modified version of BK-SVD for head approximate projection which uses the special rank-one structures involved in the calculation of the gradients. We call this method *Modified BK-SVD*, or MBK-SVD. The basic idea is to *implicitly* evaluate each Krylov-subspace iteration within BK-SVD, and avoid any explicit calculation of the adjoint operator $\mathcal{A}^*$ applied to the current estimate. The pseudocode as well as the running time analysis of MBK-SVD is deferred in [18]. Hence, we have the following theorem:

Theorem 6. *AP-ROM (with modified BK-SVD) runs in time $K = O\left(p^2 r^4 \log^2(\frac{1}{\epsilon})\text{polylog}(p)\right)$.*

Discussion. In this paper, we study provable learning of shallow polynomial network through matrix estimation problem. From this point of view, it seems plausible that the matrix-based techniques of this paper can be extended to learn networks with similar polynomial-like activation functions (such as the squared ReLU). Moreover, similar algorithms can be plausibly used to train multi-layer networks using a greedy (layer-by-layer) learning strategy. Finally, it will be interesting to integrate our methods with practical approaches such as stochastic gradient descent (SGD).

References

[1] R. Livni, S. Shalev-Shwartz, and O. Shamir. On the computational efficiency of training neural networks. In *Adv. Neural Inf. Proc. Sys. (NIPS)*, pages 855–863, 2014.

[2] M. Lin and J. Ye. A non-convex one-pass framework for generalized factorization machine and rank-one matrix sensing. In *In Adv. Neural Inf. Proc. Sys. (NIPS)*, pages 1633–1641, 2016.

[3] M. Janzamin, H. Sedghi, and A. Anandkumar. Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. *arXiv preprint arXiv:1506.08473*, 2015.

[4] Y. Tian. Symmetry-breaking convergence analysis of certain two-layered neural networks with relu nonlinearity. *In Submitted to ICLR 2017*, 2016.

[5] K. Zhong, Z. Song, P. Jain, P. L Bartlett, and I. Dhillon. Recovery guarantees for one-hidden-layer neural networks. *Proc. Int. Conf. Machine Learning*, 2016.

[6] M. Soltanolkotabi, Adel J., and J. Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *arXiv preprint arXiv:1707.04926*, 2017.

[7] Y. Li and Y. Yuan. Convergence analysis of two-layer neural networks with relu activation. *In Adv. Neural Inf. Proc. Sys. (NIPS)*, 2017.

[8] M. Fazel. *Matrix rank minimization with applications*. PhD thesis, PhD thesis, Stanford University, 2002.

[9] B. Recht, M. Fazel, and P. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

[10] T. Cai and A. Zhang. Rop: Matrix recovery via rank-one projections. *Ann. Stat.*, 43(1):102–138, 2015.

[11] Y. Chen, Y. Chi, and A. Goldsmith. Exact and stable covariance estimation from quadratic sampling via convex programming. *IEEE Trans. Inform. Theory*, 61(7):4034–4059, 2015.

[12] J. Cai, E. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

[13] K. Zhong, P. Jain, and I. Dhillon. Efficient matrix sensing using rank-1 gaussian measurements. In *Int. Conf on Algorithmic Learning Theory*, pages 3–18. Springer, 2015.

[14] S. Shalev-Shwartz, A. Gonen, and O. Shamir. Large-scale convex minimization with a low-rank constraint. In *Proc. Int. Conf. Machine Learning*, pages 329–336, 2011.

[15] C. Musco and C. Musco. Randomized block krylov methods for stronger and faster approximate singular value decomposition. In *Adv. Neural Inf. Proc. Sys. (NIPS)*, pages 1396–1404, 2015.

[16] C. Hegde, P. Indyk, and L. Schmidt. Fast recovery from a union of subspaces. In *Adv. Neural Inf. Proc. Sys. (NIPS)*, pages 4394–4402, 2016.

[17] C. Hegde, P. Indyk, and L. Schmidt. Approximation algorithms for model-based compressive sensing. *IEEE Trans. Inform. Theory*, 61(9):5129–5147, 2015.

[18] M. Soltani and C. Hegde. Towards provable learning of polynomial neural networks using low-rank matrix estimation. Available at https://www.ece.iastate.edu/ msoltani/ NIPSWS17.pdf, 2017.

[19] K. L Clarkson and D. Woodruff. Low rank approximation and regression in input sparsity time. In *Proc. ACM Symp. Theory of Comput.*, pages 81–90. ACM, 2013.

[20] M. Mahoney and P. Drineas. Cur matrix decompositions for improved data analysis. *Proc. Natl. Acad. Sci.*, 106(3):697–702, 2009.

[21] V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM J. Matrix Anal. Appl.*, 31(3):1100–1124, 2009.

[22] Z. Allen-Zhu and Y. Li. Lazysvd: Even faster svd decomposition yet without agonizing pain. In *Adv. Neural Inf. Proc. Sys. (NIPS)*, pages 974–982, 2016.