A faster and simpler algorithm for learning shallow networks

Sitan Chen SITAN@SEAS.HARVARD.EDU
Harvard
Shyam Narayanan SHYAMSN@MIT.EDU
MIT

Editors: Shipra Agrawal and Aaron Roth

Abstract

We revisit the well-studied problem of learning a linear combination of k ReLU activations given labeled examples drawn from the standard d-dimensional Gaussian measure. Chen et al. (2023) recently gave the first algorithm for this problem to run in $\mathrm{poly}(d, 1/\varepsilon)$ time when $k = O(1)$, where ε is the target error. More precisely, their algorithm runs in time $(d/\varepsilon)^{\mathrm{quasipoly}(k)}$ and learns over multiple stages. Here we show that a much simpler one-stage version of their algorithm suffices, and moreover its runtime is only $(dk/\varepsilon)^{O(k^2)}$.

Keywords: PAC learning, neural networks, multi-index models, method of moments, tensors

1. Introduction

We consider the well-studied problem of PAC learning one-hidden-layer ReLU networks from Gaussian examples. Here, there are unknown *weight vectors* $u_1, \dots, u_k \in \mathbb{S}^{d-1}$ and *output weights* $\lambda_1, \dots, \lambda_k \in \mathbb{R}$, and we are given labeled examples $(x_1, f(x_1)), \dots, (x_N, f(x_N))$ for

$$f(x) \triangleq \sum_{i=1}^{k} \lambda_i \,\mathsf{relu}(\langle u_i, x\rangle)\,, \tag{1}$$

where $x_1, \dots, x_N$ are drawn i.i.d. from the standard d-dimensional Gaussian measure γ. The goal is to output some estimator $\widehat{f}$ for which

$$\|f - \widehat{f}\|_{L_2(\gamma)} \le \varepsilon\,, \tag{2}$$

for some target error ε. In order for this to be scale-invariant, we adopt the standard (and necessary) normalization convention of assuming that $\sum_i |\lambda_i| \le \mathcal{R}$ for some parameter $\mathcal{R} \ge 1$.

This problem has been a fruitful testbed both for proving rigorous guarantees on training neural networks with gradient descent, and for developing new provably correct algorithms for nonconvex regression in high dimensions. While it has been the subject of a long line of work (Janzamin et al., 2015; Sedghi et al., 2016; Bakshi et al., 2019; Ge et al., 2018b,a; Gao et al., 2018; Diakonikolas et al., 2020b; Zhang et al., 2016; Goel et al., 2017; Daniely, 2017; Goel and Klivans, 2019; Zhong et al., 2017; Li and Yuan, 2017; Vempala and Wilmes, 2019; Zhang et al., 2019; Soltanolkotabi, 2017; Zhang et al., 2017; Diakonikolas et al., 2020a; Li et al., 2020; Goel et al., 2018; Allen-Zhu et al., 2019; Chen et al., 2022; Diakonikolas and Kane, 2020; Chen et al., 2023), it remains open to find a $\mathrm{poly}(k, d, \mathcal{R}/\varepsilon)$ time algorithm for this problem without making any additional assumptions on the network parameters. For a more thorough overview of related work, we refer the reader to the discussion in Chen et al. (2022, 2023).

Recently, Chen et al. (2023) gave the first $\mathrm{poly}(d, \mathcal{R}/\varepsilon)$ time algorithm for this problem in the regime where $k = O(1)$. Unfortunately, their dependence on k was rather large, namely $(dk\mathcal{R}/\varepsilon)^{\mathrm{quasipoly}(k)}$. In this work, we obtain the following improvement:

Theorem 1 *There is an algorithm for PAC learning one-hidden-layer ReLU networks from Gaussian examples with runtime and sample complexity* $(dk\mathcal{R}/\varepsilon)^{O(k^2)}$.

In Diakonikolas et al. (2020b) (see also Goel et al. (2020)), it was shown, roughly speaking, that any *correlational statistical query* (CSQ) algorithm time at least $d^{\Omega(k)}$ to learn to constant error in this setting. In particular, this lower bound applies to the standard approach in practice of running noisy gradient descent with respect to square loss. The algorithm we use is also a CSQ algorithm, and our Theorem 1 can thus be interpreted as saying that the lower bound of Diakonikolas et al. (2020b) is qualitatively tight, up to the particular polynomial dependence on k in the exponent. In fact, as we discuss in Remark 17 at the end of §4, for the specific hard instance constructed in the lower bound of Diakonikolas et al. (2020b), the dependence on k in the exponent that our algorithm achieves can actually be improved from quadratic to linear.

Comparison to Chen et al. (2023). Our algorithm can be thought of as a simplification of the algorithm proposed by Chen et al. (2023) in the following sense.

The starting point for their algorithm was to form empirical estimates of the moment tensors $T_\ell = \sum_i \lambda_i u_i^{\otimes \ell}$ for various choices of ℓ and contract these along a random direction $g \in \mathbb{S}^{d-1}$ into matrices $M_\ell \triangleq T_\ell(g, \dots, g, :, :) = \sum_i \lambda_i \langle u_i, g\rangle^{\ell-2} u_i u_i^\intercal$. Intuitively, these matrices constitute different linear combinations of the projectors $u_i u_i^\intercal$, and if we take enough different choices of ℓ, these matrices will collectively span the subspace $\mathrm{span}(u_1 u_1^\intercal, \dots, u_k u_k^\intercal)$. So in principle, by taking a suitable linear combination $\sum_\ell \alpha_\ell M_\ell$ of these matrices and computing its top-k singular subspace, we can get access to the subspace spanned by the weight vectors, and then exhaustively enumerate over an epsilon-net over this to find a suitable approximation to the unknown function f.

Unfortunately, as noted in Chen et al. (2023), there are a host of technical hurdles that arise in implementing this strategy, because there might be some weight vectors that are arbitrarily close to each other. *A priori*, this means that for any suitable choice of coefficients $\{\alpha_\ell\}$, some of the α_ℓ's would have to be very (in fact, arbitrarily) large, which would require estimating the moment tensors to arbitrarily small precision.

Their workaround was to argue that if one takes the top-k singular subspaces of sufficiently many different M_ℓ's and computes their joint span U, this space will contain some weight vectors. One can then subtract these from the unknown function and recurse. Unfortunately, the error in estimating weight vectors in each stage compounds exponentially in k, and under their analysis, $\Theta(\log k)$ rounds of recursion are needed, which ultimately leads to their $(d/\varepsilon)^{\mathrm{quasipoly}(k)}$ runtime.

In the present work, we show that this multi-stage approach is unnecessary, and in fact all of the information needed to reconstruct f is present in the subspace U computed in the first round of their algorithm.[1] The central ingredient in our analysis is a univariate polynomial construction (Lemma 7) that shows, roughly speaking, that for any cluster $S \subseteq [k]$ of weight vectors which are $\mathrm{poly}(1/d, 1/k, \varepsilon/\mathcal{R})$-far from all other weight vectors, there exists a linear combination of M_ℓ's which is equal to $\sum_{i \in S} \lambda_i u_i u_i^\intercal$. Crucially, the coefficients in this linear combination can be upper bounded by a quantity depending only on $d, k, \mathcal{R}/\varepsilon$ and not on the distances between the weight vectors. These linear combinations certify that U contains a vector close to each such cluster (Corollary 16), and we show (Lemma 9) that these vectors are enough to approximate f.

1. In fact, we show that it is present even in a certain low-dimensional approximation to this subspace. For technical reasons, it is essential to work with this approximation instead of the full subspace in order to get the claimed $(k/\varepsilon)^{k^2}$ dependence in Theorem 1, as opposed to a $(k/\varepsilon)^{k^4}$ dependence.

Independent Work. Diakonikolas and Kane (2023), in a concurrent work, also obtained an algorithm for learning one-hidden-layer ReLU networks from Gaussian examples. Their sample complexity and runtime is $(dk/\varepsilon)^{O(k)}$, which has an improved exponential dependence on k compared to ours, which is exponential in k^2. However, their algorithm is *improper*, meaning it cannot output a hypothesis that is also a linear combination of ReLUs, whereas our algorithm is *proper*.

2. Preliminaries

Notation. Given a positive integer k, we use $[k]$ to denote the set of integers $\{1, 2, \dots, k\}$. Likewise, given two positive integers $b \geq a$, we use $[a : b]$ to denote the set of integers $\{a, a+1, \dots, b\}$.

Given functions $a, b : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, we use $a = O(b)$ and $a \lesssim b$ interchangeably to denote that there exists an absolute constant C such that $a(z) \leq C \cdot b(z)$ for all z sufficiently large.

Given any function $f : \mathbb{R}^d \to \mathbb{R}$ and a distribution γ over $\mathbb{R}^d$, we write $\|f\|_{L_2(\gamma)} = \sqrt{\mathbb{E}_{x\sim\gamma}[f(x)^2]}$, recalling that γ denotes the standard d-dimensional normal distribution $\mathcal{N}(0, \mathbb{1})$.

We will always use u_i to denote a vector in the d-dimensional unit sphere $\mathbb{S}^{d-1}$, and λ_i, μ_i to denote real-valued scalars. For a vector v, we use $\|v\|$ to denote its ℓ_2 norm (or Euclidean norm), and $\|v\|_1$ to denote its ℓ_1 norm. Given a real symmetric matrix M, we use $\|M\|_{\mathsf{op}}$ to denote its operator norm, and $\|M\|_F$ to denote its Frobenius norm.

2.1. ReLU networks

Lemma 2 (Lemma 2.1 in Chen et al. (2023)) *Given $f = \sum_{i=1}^k \mu_i \,\mathsf{relu}(\langle u_i, \cdot\rangle)$, there exist $w \in \mathbb{R}^d$ and $\lambda_1, \dots, \lambda_k \in \mathbb{R}$ such that $f = \langle w, \cdot\rangle + \sum_{i=1}^k \lambda_i \,|\langle u_i, \cdot\rangle|$.*

In light of Lemma 2, given $w \in \mathbb{R}^d$ and $(\lambda_1, u_1), \dots (\lambda_k, u_k) \in \mathbb{R} \times \mathbb{S}^{d-1}$, let

$$f_{w,\lambda,\mathbf{u}}(x) \triangleq \langle w, x\rangle + \sum_{i=1}^k \lambda_i \,|\langle u_i, x\rangle| \,. \tag{3}$$

We use the following bound relating parameter closeness to $L_2(\gamma)$-closeness for such functions.

Proposition 3 (Lemma 3.3, restated, in Chen et al. (2021)) *For $x \sim \gamma$ and any unit vectors u, u',*

$$\mathbb{E}[(|\langle u, x\rangle| - |\langle u', x\rangle|)^2] \lesssim \|u - u'\|^2 \,. \tag{4}$$

2.2. Moment tensors

Given $g \in \mathbb{S}^{d-1}$ and $\ell \in \mathbb{N}$, define

$$T_\ell(\{\lambda_i, u_i\}) \triangleq \sum_{i=1}^k \lambda_i u_i^{\otimes \ell} \qquad \text{and} \qquad M_\ell^g(\{\lambda_i, u_i\}) \triangleq \sum_{i=1}^k \lambda_i \langle u_i, g\rangle^{\ell-2} u_i u_i^\mathsf{T} \,, \tag{5}$$

noting that the latter can be obtained by contracting the former along the direction g in the first $\ell - 2$ modes, which we denote by $M_\ell^g(\{\lambda_i, u_i\}) = T_\ell(\{\lambda_i, u_i\})(g, \cdots, g, :, :)$. When g and $\{\lambda_i, u_i\}$ are clear from context, we denote these by T_ℓ and M_ℓ respectively.

These objects can be estimated from samples as follows. Let $\mathrm{He}_\ell(\cdot)$ denote the degree-ℓ probabilist's Hermite polynomial. The polynomials $\{\frac{1}{\sqrt{\ell!}}\mathrm{He}_\ell\}_{\ell\geq 1}$ form an orthonormal basis for the space

of functions which are square-integrable with respect to γ. Define the *normalized Hermite tensor* $S_\ell : \mathbb{R}^d \to (\mathbb{R}^d)^{\otimes \ell}$ to be the tensor-valued function whose $(i_1, \ldots, i_\ell)$-th entry, given input $x \in \mathbb{R}^d$, is $\prod_{j=1}^d \frac{1}{\sqrt{\ell_j!}} \mathrm{He}_{\ell_j}(x_j)$, where ℓ_j is the number of occurrences of j within $(i_1, \ldots, i_\ell)$.

Lemma 4 (Lemma 4.2 in Chen et al. (2023)) *Let $\ell \in \{1, 2, 4, 6, \ldots\}$ and*

$$C_\ell \triangleq \begin{cases} 1/2 & \text{if } \ell = 1 \\ \frac{\mathrm{He}_\ell(0) + \ell \mathrm{He}_{\ell-2}(0)}{\sqrt{2\pi \ell!}} & \text{if } \ell \text{ even} \end{cases} \tag{6}$$

Let $\eta > 0$. Given samples $\{(x_i, f_{w,\lambda,\mathbf{u}}(x_i)\}_{i \in 1,\ldots,N}$ for $x_i \sim \gamma$ and $N \geq \ell^{O(\ell)} d^{2\ell} \mathcal{R}^2 / \eta^2$, with high probability the tensor

$$\widehat{T} = \frac{1}{2C_\ell N} \sum_i f_{w,\lambda,\mathbf{u}}(x_i) \cdot S_\ell(x_i) \tag{7}$$

satisfies $\|\widehat{T} - T_\ell\|_F \leq \eta$ if ℓ is even, and otherwise satisfies $\|\widehat{T} - w\|_2 \leq \eta$ if $\ell = 1$. In particular, for even ℓ, if we define $\widehat{M}_\ell^g \triangleq \widehat{T}_\ell(g, \cdots, g, :, :)$ then $\|\widehat{M}_\ell^g - M_\ell^g\|_F \leq \eta$.

When g and N are clear from context, we will use $\widehat{M}_\ell$ to refer to the empirical estimate $\widehat{M}_\ell^g$ which is obtained using N samples. We will also use $\widehat{w}$ to refer to $\widehat{T}$ when $\ell = 1$ to emphasize that it is an empirical estimate of the linear component w in $f_{w,\lambda,\mathbf{u}}$.

2.3. Random contraction

As in Chen et al. (2023), our algorithm is based on extracting information about the parameters of the network from $\{M_\ell^g\}$ for a random choice of unit vector g. The randomness in g ensures that with high probability, any two weight vectors u_i, u_j are close/far if and only if their projections $\langle u_i, g \rangle, \langle u_j, g \rangle$ are as well. Formally:

Lemma 5 (Lemma 2.2 in Chen et al. (2023)) *With probability at least $4/5$ over random $g \in \mathbb{S}^{d-1}$, for all i, j and $\sigma \in \{\pm 1\}$,*

$$\frac{c}{\sqrt{d}} \cdot \frac{1}{k^2} \leq \frac{|\langle u_i + \sigma u_j, g \rangle|}{\|u_i + \sigma u_j\|} \leq \frac{c'}{\sqrt{d}} \cdot \sqrt{\log k} \tag{8}$$

for some absolute constants $c, c' > 0$.

Henceforth, we condition on the event that g satisfies Lemma 5. We will denote

$$z_i \triangleq \langle u_i, g \rangle \tag{9}$$

and, because of the absolute values in the definition of $f_{w,\lambda,\mathbf{u}}$, we may assume without loss of generality that

$$0 \leq z_1 \leq \cdots \leq z_k \,. \tag{10}$$

2.4. Estimating test error

Our algorithm will produce a list of many candidate estimates, at least one of which is guaranteed to be sufficiently close in $L_2(\gamma)$ to $f_{w,\lambda,\mathbf{u}}$. In order to identify an estimate from the list with this property, we use the following standard result on validation:

Lemma 6 *Let $F : \mathbb{R}^d \to \mathbb{R}$ be a $2\mathcal{R}$-Lipschitz one-hidden-layer ReLU network of size at most $2k$. Let $\delta > 0$, and let $\upsilon > 0$ be a parameter satisfying $\upsilon^2 < 4\mathcal{R}^2 k$. Given $N \gtrsim k^2(\mathcal{R}/\upsilon)^4 \log(1/\delta)$ samples $x_1, \dots, x_N \sim \gamma$, we have that*

$$\left|\mathop{\mathbb{E}}_{\gamma}[F^2] - \frac{1}{N}\sum_{i=1}^{N} F(x_i)^2\right| \le \upsilon^2. \tag{11}$$

We will ultimately take F to be $f_{w,\lambda,\mathbf{u}} - \widehat{f}$ for various $\widehat{f}$ in our list of estimates. All of the $\widehat{f}$ we consider will be $\mathcal{R}$-Lipschitz and have size at most k, so $F = f_{w,\lambda,\mathbf{u}} - \widehat{f}$ will satisfy the hypotheses of Lemma 6.

3. Polynomial construction

A key tool in establishing our main result is the following polynomial construction.

Lemma 7 *Let $0 < \Delta < 1$, and let $-1 \le x_1 < \cdots < x_k \le 1$. Suppose there are indices $1 \le a < b \le k$ such that $x_{b+1} > x_b + \Delta$ and $x_a > x_{a-1} + \Delta$. Then, there exists a degree (at most) k^2 polynomial p with coefficients bounded by $O(1/\Delta)^{O(k^2)}$ such that*

$$p(x_s) = \mathbb{1}[a \le s \le b] \tag{12}$$

for all $s \in [k]$.

Proof Define $I = [a : b]$ to be the set of indices between a and b, inclusive. We consider the polynomial

$$p(x) = \prod_{j \notin I}\Big(1 - \prod_{i \in I} \frac{x - x_i}{x_j - x_i}\Big). \tag{13}$$

It is clear that the degree of this polynomial is at most $|I| \cdot (k - |I|) \le k^2$. Next, because every $x_i \in [-1, 1]$ and every $|x_j - x_i| \ge \Delta$, it is clear that $\prod_{j \in I} \frac{x - x_i}{x_j - x_i}$ has all coefficients bounded by $O(1/\Delta)^k$, which means the full polynomial $p(x)$ has all coefficients bounded by $O(1/\Delta)^{k^2}$.

Next, we evaluate this polynomial on x_s, where $s \in I$. In this case, note that $\prod_{i \in I} \frac{x_s - x_i}{x_j - x_i} = 0$ for any $j \notin I$, because when we set $i = s$, the fraction is 0. Therefore, $1 - \prod_{i \in I} \frac{x_s - x_i}{x_j - x_i} = 1$ for all $j \notin I$, so $p(x_s) = \prod_{j \notin I} 1 = 1$. Finally, we evaluate this polynomial on x_s, where $s \notin I$. Note that for $j = s$, $\prod_{i \in I} \frac{x_s - x_i}{x_j - x_i} = 1$, so $1 - \prod_{i \in I} \frac{x_s - x_i}{x_j - x_i} = 0$ for $j = s$. Therefore, $p(x_s) = 0$, because one of the terms in the product that comprises p evaluates to 0. ■

Lemma 7 will end up being applied on $\langle u_i, g\rangle^2$ for some random vector g. Using linear combinations of the matrices $\widehat{M}_\ell$ described in §2.2, we can estimate $\sum \lambda_i p(\langle u_i, g\rangle^2) u_i u_i^\intercal$ for any polynomial p. Lemma 7 allows us to choose a polynomial p that isolates out a "cluster" of somewhat close vectors $\{u_i\}_{i \in I}$, as long as the remaining vectors u_j (for $j \notin I$) are of distance at least Δ away. Hence, the linear combination of the matrices $\widehat{M}_\ell$ corresponding to this choice of p will result in a matrix which closely approximates the direction of u_i for $i \in I$.

Algorithm 1: NETLEARN(f, ε)

Input: Sample access to unknown one-hidden-layer network $f_{w,\lambda,\mathbf{u}}$, target error ε

Output: ε-close estimate $\widehat{f}$ for $f_{w,\lambda,\mathbf{u}}$

$\tau \leftarrow \frac{C\varepsilon}{k}$, $\xi \leftarrow \frac{C\varepsilon}{k\mathcal{R}}$, $\Delta \leftarrow \frac{C^2\xi^2\tau\varepsilon}{2k^4d^{3/2}\mathcal{R}}$, $\eta' = \nu \leftarrow \frac{C^2\xi^2\tau\Delta^{O(k^2)}}{d\cdot\mathcal{R}}$, $\eta = (\eta')^2$.

$N \leftarrow (kd)^{O(k^2)}\mathcal{R}^2/\eta^2$, $N_{\mathsf{val}} \leftarrow O((k\mathcal{R}/\varepsilon)^4\log(k\mathcal{R}/\varepsilon))$.

Form estimates $\{\widehat{M}_\ell\}_{\ell=2,4,\ldots,2k^2+2}$ and $\widehat{w}$ from N samples $\{(x_j, y_j)\}_{j=1}^N$. `// see end of §2.2`

For each $\ell \in \{2, 4, \ldots, 2k^2+2\}$, form the projector Π_ℓ to the span of the eigenvectors of $\widehat{M}_\ell$ with eigenvalue at least η' in absolute value.

Compute $\sum_\ell \Pi_\ell$, and let V denote the subspace spanned by the eigenvectors of $\sum_\ell \Pi_\ell$ with eigenvalue at least ν.

Construct a $\xi/2$-net N_{u} in Euclidean distance over the set of vectors of unit norm in V.

Construct a ξ-net N_λ over the interval $[-\mathcal{R}, \mathcal{R}]$.

Draw N_{val} additional samples $\{(x_j, y_j)\}_{j=N+1}^{N+N_{\mathsf{val}}}$.

for $m = 0, 1, \ldots, k$ **do**
  for $\widehat{\lambda}_1, \ldots, \widehat{\lambda}_m \in N_\lambda$ *and* $\widehat{u}_1, \ldots, \widehat{u}_m \in N_{\mathsf{u}}$ **do**
    if $\frac{1}{N}\sum_{j=N+1}^{N+N_{\mathsf{val}}}(y_j - f_{\widehat{w},\widehat{\lambda},\widehat{u}})^2 \le \varepsilon^2/2$ **then**
      return $f_{\widehat{w},\widehat{\lambda},\widehat{u}}$.
    end
  end
end
return *Fail*

4. Algorithm and analysis

Here we give an analysis for our main algorithm, NETLEARN, the full specification of which is given in Algorithm 1. Roughly speaking, the algorithm proceeds by forming empirical estimates $\widehat{M}_\ell$ for the moment matrices M_ℓ defined in Eq. (5) for $\ell \le O(k^2)$, computing the top singular subspaces of the various $\widehat{M}_\ell$, finding an $O(k)$-dimensional approximation V to the collective span of these subspaces, and finally brute-forcing over V to find a sufficiently good estimate for $f_{w,\lambda,\mathbf{u}}$.

In the algorithm and analysis, we have several important parameters: $\eta, \eta', \nu, \Delta, \xi$, and τ. We will not set the exact values of these parameters in the analysis until the end, but we will assume that $\eta \le \eta', \nu \le \Delta \le \xi, \tau \le \varepsilon \le 1$, where we recall that ε is our desired accuracy.

In §4.1, we introduce some conventions for handling weight vectors which are closely spaced by dividing them up in our analysis into clusters. In §4.2 we give the main part of our analysis in which we argue that the net N_{u} constructed in NETLEARN contains vectors close to a subset of weight vectors of the unknown network $f_{w,\lambda,\mathbf{u}}$ that could be used to approximate $f_{w,\lambda,\mathbf{u}}$ to sufficiently small error. We conclude the proof of Theorem 1 in §4.5.

4.1. Basic clustering

A key challenge in learning one-hidden-layer networks without making any assumptions on the weight vectors is that parameter recovery is impossible, because there may exist weight vectors in

the network which are arbitrarily close to each other. In Chen et al. (2023), the authors addressed this by giving a rather delicate clustering-based argument based on grouping together weight vectors that were close at multiple different scales.

In this work, we sidestep this multi-scale analysis and show that under a fixed scale, a naive clustering of the weight vectors suffices for our analysis. Indeed, for a scale $\Delta > 0$ to be tuned later, let $I_1 \sqcup \cdots \sqcup I_m$ be a partition of $[k]$ into disjoint, contiguous intervals such that any adjacent z_i, z_{i+1} in the same interval are at most Δ-apart, whereas the distance between the endpoints of any two intervals exceeds Δ. (Recall that $z_i \triangleq \langle u_i, g\rangle$, where g is a randomly chosen unit vector, and that we assume the indices are sorted in increasing order of z_i.) We remark that $I_1, \ldots, I_m$ are only referenced in the analysis, and our actual algorithm does not need to know this partition.

Note that under this partition, any two $z_i, z_{i'}$ in the same interval are at most $k\Delta$-apart. Recalling that we are conditioning on the event of Lemma 5, this implies that for such i, i',

$$\|u_i - u_{i'}\| \lesssim \Delta \cdot k^3\sqrt{d}\,. \tag{14}$$

In every interval I_j, let i_j^* denote its left endpoint. Also define

$$\overline{\lambda}_j \triangleq \sum_{i \in I_j} \lambda_i\,. \tag{15}$$

For a threshold τ to be tuned later, define

$$J_{\mathsf{big}} \triangleq \{j \in [m] : |\overline{\lambda}_j| > \tau\}\,. \tag{16}$$

Intuitively, J_{big} corresponds to clusters of neurons which are learnable, as the neurons coming from those clusters do not "cancel" significantly with each other.

The following shows that a linear combination of projectors to weight vectors from the same cluster is well-approximated by a projector to a single weight vector in that cluster:

Proposition 8 *For any $j \in [m]$, and any fixed $i' \in I_j$,*

$$\left\|\overline{\lambda}_j u_{i'} u_{i'}^\mathsf{T} - \sum_{i \in I_j} \lambda_i u_i u_i^\mathsf{T}\right\|_{\mathsf{op}} \lesssim \Delta \cdot k^3\sqrt{d}\,\|\lambda\|_1\,. \tag{17}$$

Proof First, note that by triangle inequality, $\|u_{i'}u_{i'}^\mathsf{T} - u_i u_i^\mathsf{T}\|_{\mathsf{op}} \le \|u_{i'}u_{i'}^\mathsf{T} - u_i u_{i'}^\mathsf{T}\|_{\mathsf{op}} + \|u_i u_{i'}^\mathsf{T} - u_i u_i^\mathsf{T}\|_{\mathsf{op}} = 2\|u_{i'} - u_i\|$, since u_i and $u_{i'}$ are both unit vectors. Hence, for any $i \in I_j$, we have

$$\|u_{i'}u_{i'}^\mathsf{T} - u_i u_i^\mathsf{T}\|_{\mathsf{op}} \le 2\|u_{i'} - u_i\| \lesssim \Delta \cdot k^3\sqrt{d} \tag{18}$$

by Eq. (14), from which the claim follows by triangle inequality. ■

Next, we show that to learn the nonlinear parts of $f_{w,\lambda,\mathbf{u}}$, it suffices to estimate $\overline{\lambda}_j$ and $u_{i_j^*}$ for clusters $j \in J_{\mathsf{big}}$.

Lemma 9 *Let $\varepsilon > 0$. For sufficiently small constants $c_1, c_2, c_3 > 0$, suppose*

$$\tau \le \frac{c_1\varepsilon}{k},\quad \xi \le \frac{c_2\varepsilon}{k \cdot \max(1, \|\lambda\|_1)},\quad \Delta \le \frac{c_3\varepsilon}{k^4\sqrt{d}\,\|\lambda\|_1}\,. \tag{19}$$

If the parameters $\{\widehat{\lambda}_j, \widehat{u}_j\}_{j \in J_{\mathsf{big}}} \in \mathbb{S}^{d-1} \times \mathbb{R}$ *satisfy* $|\widehat{\lambda}_j - \overline{\lambda}_j| \le \xi$ *and* $\|\widehat{u}_j - u_{i_j^*}\| \le \xi$ *for all* $j \in J_{\mathsf{big}}$*, then*

$$\Big\|\sum_{i=1}^{k} \lambda_i \left|\langle u_i, \cdot\rangle\right| - \sum_{j \in J_{\mathsf{big}}} \widehat{\lambda}_j \left|\langle \widehat{u}_j, \cdot\rangle\right|\Big\|_{L_2(\gamma)} \le \varepsilon \tag{20}$$

Proof For every j (including $j \notin J_{\mathsf{big}}$), we have by triangle inequality and Proposition 3,

$$\begin{aligned}
\Big\|\sum_{i \in I_j} \lambda_i \left|\langle u_i, \cdot\rangle\right| - \overline{\lambda}_j \left|\langle \widehat{u}_j, \cdot\rangle\right|\Big\|_{L_2(\gamma)} &\le \sum_{i \in I_j} |\lambda_i| \cdot \Big\| |\langle u_i, \cdot\rangle| - |\langle \widehat{u}_j, \cdot\rangle| \Big\|_{L_2(\gamma)} && (21)\\
&\lesssim \sum_{i \in I_j} |\lambda_i| \cdot (\|u_i - u_{i_j^*}\| + \|u_{i_j^*} - \widehat{u}_j\|) && (22)\\
&\lesssim \|\lambda\|_1 \cdot \left(\Delta \cdot k^3\sqrt{d} + \|u_{i_j^*} - \widehat{u}_j\|\right) && (23)\\
&\le \|\lambda\|_1 \cdot (\Delta \cdot k^3\sqrt{d} + \xi)\,. && (24)
\end{aligned}$$

Furthermore, for all $j \in J_{\mathsf{big}}$,

$$\big\|\overline{\lambda}_j \left|\langle \widehat{u}_j, \cdot\rangle\right| - \widehat{\lambda}_j \left|\langle \widehat{u}_j, \cdot\rangle\right|\big\|_{L_2(\gamma)} = |\overline{\lambda}_j - \widehat{\lambda}_j| \cdot \big\||\langle \widehat{u}_j, \cdot\rangle|\big\|_{L_2(\gamma)} \lesssim \xi\,. \tag{25}$$

and for all $j \notin J_{\mathsf{big}}$,

$$\|\overline{\lambda}_j \left|\langle u_{i_j^*}, \cdot\rangle\right|\|_{L_2(\gamma)} \lesssim |\overline{\lambda}_j| \le \tau. \tag{26}$$

By triangle inequality and the fact that $m \le k$, we conclude that

$$\Big\|\sum_{i=1}^{k} \lambda_i \left|\langle u_i, \cdot\rangle\right| - \sum_{j \in J_{\mathsf{big}}} \widehat{\lambda}_j \left|\langle \widehat{u}_j, \cdot\rangle\right|\Big\| \lesssim \|\lambda\|_1 \cdot (\xi k + \Delta \cdot k^4\sqrt{d}) + k(\xi + \tau)\,, \tag{27}$$

and the lemma follows from the bounds in Eq. (19). ■

4.2. Analysis of PCA: Overview

Let η be a parameter to be tuned later. By Lemma 4, using $N = k^{O(k^2)} d^{O(k^2)} \mathcal{R}^2/\eta^2$ samples, we can form an empirical estimate $\widehat{M}_\ell$ for which $\|\widehat{M}_\ell - M_\ell\|_F \le \eta$, for any positive even $\ell \le O(k^2)$. (We assume WLOG that $\widehat{M}_\ell$ is symmetric.) In Line 1 of NETLEARN, we do this for all $\ell \in \{2, 4, \ldots, 2k^2 + 2\}$.

For each $\widehat{M}_\ell$, we can decompose it as

$$\widehat{M}_\ell = \sum_{i=1}^{d} \rho_i^{(\ell)} w_i^{(\ell)} (w_i^{(\ell)})^\mathsf{T}\,, \tag{28}$$

where $\rho_i^{(\ell)} \in \mathbb{R}$ and $w_i^{(\ell)} \in \mathbb{S}^d$ are the eigenvalues and eigenvectors, respectively, of $\widehat{M}_\ell$. In Line 1 of NETLEARN, we compute Π_ℓ as the projection to the span of the eigenvectors with eigenvalue at least η' in absolute value, i.e.,

$$\Pi_\ell = \sum_{i : |\rho_i^{(\ell)}| \ge \eta'} w_i^{(\ell)} (w_i^{(\ell)})^\mathsf{T}\,. \tag{29}$$

Next, in Line 1 of NETLEARN, we compute $\sum_\ell \Pi_\ell$, which we can decompose as

$$\sum_\ell \Pi_\ell = \sum_{i=1}^{d} \kappa_i v_i v_i^\intercal \,, \tag{30}$$

where κ_i, v_i are the eigenvalues and eigenvectors, respectively, of $\sum_\ell \Pi_\ell$. We pick V as the span of v_i with $|\kappa_i| \geq \nu$.

In analyzing PCA, we have two main steps. First, in §4.3 we show that V has low dimension. This is because we wish to brute force over choices of $\widehat{u}_1, \ldots, \widehat{u}_m$ in V to find a suitable set of directions. Next, in §4.4 we show that every u_i, where $i \in j$ for $j \in J_{\mathsf{big}}$, is close to V. This will allow us to prove that there exists an approximate solution in our brute force search.

4.3. V has low dimension

Consider any fixed $\ell \in \{2, 4, \ldots, 2k^2 + 2\}$, and consider the empirical estimate $\widehat{M}_\ell$ for which $\|\widehat{M}_\ell - M_\ell\|_F \leq \eta$. To bound the dimension of V, we first show that every not-too-small eigenvector of $\widehat{M}_\ell$ (for all ℓ) is close to the span of $\{u_i\}_{i=1}^k$.

Lemma 10 *Suppose that w is a (unit) eigenvector of $\widehat{M}_\ell$ with eigenvalue at least η' in absolute value. Then, w is within Euclidean distance η/η' of the subspace span$(\{u_i\})$.*

Proof Suppose $\widehat{M}_\ell w = \rho w$ for some $|\rho| \geq \eta'$. Note that $\|M_\ell w - \widehat{M}_\ell w\| \leq \|\widehat{M}_\ell - M_\ell\|_{\mathsf{op}} \cdot \|w\| \leq \eta$, since $\|w\| = 1$. Hence, $M_\ell w$ is within η of ρw. However, note that

$$M_\ell w = \sum_{i=1}^{k} \lambda_i \langle u_i, g\rangle^{\ell-2} \cdot u_i u_i^\intercal w = \sum_{i=1}^{k} \lambda_i \langle u_i, g\rangle^{\ell-2} \cdot \langle u_i, w\rangle \cdot u_i,$$

which is in the span of $\{u_i\}_{i=1}^k$. Hence, ρw is within η of span$(\{u_i\})$, and since $|\rho| \geq \eta'$, this means w is within η/η' of span$(\{u_i\})$. ■

Let $U = \text{span}(\{u_i\})$. Let Π_U be the projection matrix onto U, and $\Pi_U^\perp$ be the projection matrix onto the orthogonal complement of U. Using Lemma 10, we can bound the inner product between $\Pi_U^\perp$ and the projection matrix Π_ℓ.

Corollary 11 *For every ℓ,* $\text{Tr}(\Pi_U^\perp \cdot \Pi_\ell) \leq d\,(\eta/\eta')^2$.

Proof First, note that $(w_i^{(\ell)})^\intercal \Pi_U^\perp w_i^{(\ell)} = \|\Pi_U^\perp w_i\|^2$, which is precisely the squared distance from $w_i^{(\ell)}$ to span$(\{u_i\})$. So, if $|\rho_i^{(\ell)}| \geq \eta'$, then $\text{Tr}(\Pi_U^\perp \cdot w_i^{(\ell)} (w_i^{(\ell)})^\intercal) = w_i^{(\ell)} \Pi_U^\perp w_i \leq (\eta/\eta')^2$ by Lemma 10. Recalling the definition of Π_ℓ in Eq. (30), we obtain the claimed bound. ■

Because there are at most $O(k^2)$ choices of ℓ, this implies that $\text{Tr}(\Pi_U^\perp \cdot \sum_\ell \Pi_\ell) \leq O(dk^2) \cdot (\eta/\eta')^2$. Now, let Π_V be the projection matrix to the subspace V. We now bound $\text{Tr}(\Pi_U^\perp \cdot \Pi_V)$.

Proposition 12 *We have that* $\text{Tr}(\Pi_U^\perp \cdot \Pi_V) \leq O(dk^2(\eta/\eta')^2/\nu)$.

Proof Recall that $\sum_\ell \Pi_\ell$ has eigendecomposition $\sum_{i=1}^d \kappa_i v_i v_i^\intercal$. Since every Π_ℓ is positive semidefinite, this means $\kappa_i \geq 0$ for all i. Moreover, $\Pi_V = \sum_{i:\kappa_i \geq \nu} v_i v_i^\intercal$, which means that $\nu \cdot \Pi_V \preccurlyeq \sum_\ell \Pi_\ell$. Therefore, $\operatorname{Tr}(\Pi_U^\perp \cdot \Pi_V) \leq \frac{1}{\nu} \cdot \operatorname{Tr}(\Pi_U^\perp \cdot \sum_\ell \Pi_\ell) \leq O(dk^2(\eta/\eta')^2/\nu)$. ■

Hence, we have the following bound on the dimension of V.

Lemma 13 *We have that* $\operatorname{Tr}(\Pi_V) \leq k + O(dk^2(\eta/\eta')^2/\nu)$. *Hence, the dimension of* V *is at most* $k + O(dk^2(\eta/\eta')^2/\nu)$.

Proof For any projection matrix to a subspace S, its trace is the same as the dimension of S. So, we just need to prove that $\operatorname{Tr}(\Pi_V) \leq k + O(dk^2(\eta/\eta')^2/\nu)$.

Note that $\operatorname{Tr}(\Pi_V) = \operatorname{Tr}(\Pi_V \cdot (\Pi_U + \Pi_U^\perp)) = \operatorname{Tr}(\Pi_V \cdot \Pi_U) + \operatorname{Tr}(\Pi_V \cdot \Pi_U^\perp)$. Since $\Pi_V, \Pi_U \preccurlyeq I$ as they are projection matrices, $\operatorname{Tr}(\Pi_V \cdot \Pi_U) \leq \operatorname{Tr}(\Pi_U) = \dim(U) \leq k$. By Proposition 12, we have that $\operatorname{Tr}(\Pi_V \cdot \Pi_U^\perp) \leq O(dk^2(\eta/\eta')^2/\nu)$. This completes the proof. ■

4.4. Each important u_i is (almost) in the span of V

In this subsection, we show that every "important" u_i (i.e., where $i \in I_j$ for $j \in J_{\mathsf{big}}$) is reasonably close to the span of this subspace V.

We recall that V is the subspace found in line 5 of Algorithm 1, and Π_V is the projection to V. We also define $\Pi_V^\perp = I - \Pi_V$ to be the projection matrix to the orthogonal complement of V.

First, we show that every M_ℓ does not have large inner product with the projection $\Pi_V^\perp$.

Lemma 14 *For all* $\ell \in \{2, 4, \ldots, 2k^2 + 2\}$, *we have that* $|\operatorname{Tr}(\Pi_V^\perp \cdot M_\ell)| \leq d \cdot (\|\lambda\|_1 \cdot \nu + O(\eta'))$.

Proof Recalling Eq. (30) and the definition of V, we have $\Pi_V^\perp = \sum_{i:\kappa_i < \nu} v_i v_i^\intercal$. So,

$$\operatorname{Tr}\Big(\Pi_V^\perp \cdot \sum_\ell \Pi_\ell\Big) = \sum_{i:\kappa_i < \nu} \kappa_i \leq d \cdot \nu\,.$$

Next, since $\Pi_V^\perp$ and Π_ℓ are both positive semidefinite, $\operatorname{Tr}(\Pi_V^\perp \cdot \Pi_\ell) \geq 0$, so for all ℓ,

$$\operatorname{Tr}(\Pi_V^\perp \cdot \Pi_\ell) \leq d \cdot \nu\,. \tag{31}$$

Note that all the eigenvalues of M_ℓ are bounded by $\|\lambda\|_1$ in absolute value since $\langle u_i, g\rangle \leq 1$. Because $\|\widehat{M}_\ell - M_\ell\|_{\mathsf{op}} \leq \|\widehat{M}_\ell - M_\ell\|_F \leq \eta$, all of the eigenvalues $\rho_i^{(\ell)}$ of $\widehat{M}_\ell$ are bounded by $\|\lambda\|_1 + \eta$ in absolute value. Recalling that Π_ℓ is the projector to the span of the eigenvectors of $\widehat{M}_\ell$ with eigenvalue of magnitude at least η', we have that

$$-(\|\lambda\|_1 + \eta) \cdot \Pi_\ell - \eta' \cdot I \preccurlyeq \widehat{M}_\ell \preccurlyeq (\|\lambda\|_1 + \eta) \cdot \Pi_\ell + \eta' \cdot I. \tag{32}$$

By combining Equations (31) and (32), and the fact that $\|M_\ell - \widehat{M}_\ell\|_{\mathsf{op}} \leq \eta$, we have that

$$\begin{aligned} |\operatorname{Tr}(\Pi_V^\perp \cdot M_\ell)| &\leq (\|\lambda\|_1 + \eta) \cdot \operatorname{Tr}(\Pi_V^\perp \cdot \Pi_\ell) + \operatorname{Tr}(\Pi_V^\perp \cdot (\eta + \eta') \cdot I) \\ &\leq d \cdot (\|\lambda\|_1 \cdot \nu + O(\eta'))\,. \end{aligned}$$

■

Next, we show that this implies that for every "important" u_i, $u_i^\intercal \Pi_V^\perp u_i$ is small, which will be essential to showing that u_i must be close to the span of V. The proof of this will crucially use the polynomial construction from Lemma 7.

Lemma 15 *Suppose that $i \in I_j$ for some $j \in J_{\text{big}}$. Then,*

$$|u_i^\intercal \Pi_V^\perp u_i| \le \frac{1}{\tau}\left[O(1/\Delta)^{O(k^2)} \cdot d \cdot (\|\lambda\|_1 \cdot \nu + \eta') + O(\|\lambda\|_1 \cdot k^3 d^{3/2} \cdot \Delta)\right]. \tag{33}$$

Proof Let $p(x) = \sum_{\ell=0}^{k^2} p_\ell x^\ell$ be the polynomial from Lemma 7 s.t. $p(\langle u_i, g\rangle^2) = \mathbb{1}[i \in I_j]$. Then,

$$\sum_{\ell=0}^{k^2} p_\ell M_{2+2\ell} = \sum_{\ell=0}^{k^2} p_\ell \cdot \sum_{i=1}^{k} \lambda_i \cdot \langle u_i, g\rangle^{2\ell} u_i u_i^\intercal = \sum_{i=1}^{k} \lambda_i \cdot \sum_{\ell=0}^{k^2} p_\ell \langle u_i, g\rangle^{2\ell} u_i u_i^\intercal = \sum_{i \in I_j} \lambda_i u_i u_i^\intercal.$$

Since $|\langle u_i, g\rangle - \langle u_{i'}, g\rangle| \ge \Delta$ for all $i \in I_j, i' \notin I_j$, and since we are assuming $\langle u_i, g\rangle, \langle u_{i'}, g\rangle \ge 0$, this implies that $|\langle u_i, g\rangle^2 - \langle u_{i'}, g\rangle^2| \ge \Delta^2$ for all $i \in I_j, i' \notin I_j$. Hence, Lemma 7 implies every coefficient $p_\ell \le O(1/\Delta^2)^{k^2} = O(1/\Delta)^{O(k^2)}$, so by Lemma 14 we have

$$\left|\operatorname{Tr}\left(\Pi_V^\perp \cdot \sum_{i \in I_j} \lambda_i u_i u_i^\intercal\right)\right| \le O(1/\Delta)^{O(k^2)} \cdot d \cdot (\|\lambda\|_1 \cdot \nu + \eta').$$

By Proposition 8, we have that

$$\left|\operatorname{Tr}(\Pi_V^\perp \cdot \overline{\lambda}_j \cdot u_i u_i^\intercal)\right| \le O(1/\Delta)^{O(k^2)} \cdot d \cdot (\|\lambda\|_1 \cdot \nu + \eta') + O(\|\lambda\|_1 \cdot k^3 d^{3/2} \cdot \Delta).$$

Since $j \in J_{\text{big}}$, this means $|\overline{\lambda}_j| \ge \tau$, implying the bound on $|u_i^\intercal \Pi_V^\perp u_i| = |\operatorname{Tr}(\Pi_V^\perp \cdot u_i u_i^\intercal)|$. ■

As a corollary, we have that u_i is close to the span of V.

Corollary 16 *For any $i \in I_j$ where $j \in J_{\text{big}}$, the distance from u_i to V is at most*

$$\tau^{-1/2}\sqrt{O(1/\Delta)^{O(k^2)} \cdot d \cdot (\|\lambda\|_1 \cdot \nu + \eta') + O(\|\lambda\|_1 \cdot k^3 d^{3/2} \cdot \Delta)}. \tag{34}$$

Proof Write $u_i = \Pi_V u_i + \Pi_V^\perp u_i$. Note that $\Pi_V u_i \in \operatorname{span}(V)$, so we just need to bound $\|\Pi_V^\perp u_i\|$. But $\Pi_V^\perp$ is a projection, so $\|\Pi_V^\perp u_i\|^2 = u_i^\intercal \Pi_V^\perp u_i$. The claim follows by Lemma 15. ■

4.5. Putting everything together

We recall that $\mathcal{R} \ge 1$ is a promised upper bound for $\|\lambda\|_1$. For small constant $C > 0$, take

$$\tau = \frac{C\varepsilon}{k}, \quad \xi = \frac{C\varepsilon}{k \cdot \mathcal{R}}, \quad \Delta = \frac{C^2\xi^2\tau \cdot \varepsilon}{2k^4 d^{3/2} \cdot \mathcal{R}}, \quad \eta' = \nu = \frac{C^2\xi^2\tau \cdot \Delta^{O(k^2)}}{d \cdot \mathcal{R}}, \quad \eta = (\eta')^2. \tag{35}$$

Under this, by Lemma 13 the dimension of V is at most $k + O(d \cdot k^2 \cdot \nu)$, since $\eta' = \nu$ and $\eta = (\eta')^2$. However, $\nu \le C^2 \cdot \xi^2/d \le C^2/(k^2 d)$. So if C is small enough, the dimension of V is at most $k + 0.1$, so is at most k. By Corollary 16, for $i \in I_j, j \in J_{\text{big}}$, every u_i has distance at most $\sqrt{\frac{O(C^2\xi^2\tau)}{\tau}} \le \frac{\xi}{4}$ to V. For C sufficiently small, we also have that τ, ξ, Δ satisfy the constraints of Lemma 9. Finally, it is straightfoward to verify that $1/\eta = (dk\mathcal{R}/\varepsilon)^{O(k^2)}$.

Now, for each $j \in J_{\text{big}}$, we recall the definition of $\bar{\lambda}_j$. Since $|\bar{\lambda}_j| \le \mathcal{R}$ by our assumption that $\sum |\lambda_i| \le \mathcal{R}$, by the definition of N_λ (see Line 1 of Algorithm 1), there exists $\widehat{\lambda}_j \in N_\lambda$ within

distance ξ of $\bar{\lambda}_j$. Next, $u_{i_j^*}$ has distance at most $\frac{\xi}{2}$ to V, and importantly, $\|u_{i_j^*} - \Pi_V u_{i_j^*}\| \le \frac{\xi}{4}$ and so $|\|\Pi_V u_{i_j^*}\| - 1| \le \frac{\xi}{4}$. So, for $u = \frac{\Pi_V u_{i_j^*}}{\|\Pi_V u_{i_j^*}\|}$, $\|u_{i_j^*} - u\| \le \frac{\xi}{2}$. Therefore, by the definition of N_{u} (see Line 1 of Algorithm 1), there exists $\widehat{u}_j$ within distance ξ of $u_{i_j^*}$.

Therefore, our algorithm will find some $m_{\mathsf{big}} = |J_{\mathsf{big}}|$ and $(\widehat{\lambda}_1, \widehat{u}_1), \dots, (\widehat{\lambda}_{m_{\mathsf{big}}}, \widehat{u}_{m_{\mathsf{big}}})$ satisfying the conditions of Lemma 9, which will thus be within ε of the true answer in the distance $\|\cdot\|_{L_2(\gamma)}$.

The runtime is dominated by the time it takes to estimate $\{M_\ell\}$ and w, which requires

$$(k^2 d)^{O(k^2)} \cdot \mathcal{R}^2/\eta^2 = (dk\mathcal{R}/\varepsilon)^{O(k^2)} \tag{36}$$

samples by Lemma 4, and the time it takes to enumerate over sets of at most k vectors from N_{u}, and over weights from N_λ, which are of size

$$|N_{\mathsf{u}}| \le O(1/\xi)^{O(k)} = O(k\mathcal{R}/\varepsilon)^{O(k)} \qquad \text{and} \qquad |N_\lambda| \le O(k\mathcal{R}/\varepsilon)^{O(k)}\,. \tag{37}$$

We remark that the net N_{u} can be algorithmically constructed by selecting random points. Indeed, for any point x on a k-dimensional sphere, a random point on the sphere is within $\xi/2$ of x with probability at least $\Omega(\xi)^k$, so for any $\xi/2$-net N_{u}^* of size $O(1/\xi)^k$, if N_{u} is constructed as $O(1/\xi)^{2k}$ random points on the sphere, then every point in N_{u}^* will be within $\xi/2$ of at least one point in N_u with high probability. So, N_u is a ξ-net of the k-dimensional unit sphere. Hence, we enumerate over at most $(|N_u| \cdot |N_\lambda|)^{O(k)} \le O(k \cdot \mathcal{R}/\varepsilon)^{O(k^2)}$ candidate solutions.

By Lemma 6, our algorithm will successfully verify any candidate solution $\{(\widehat{\lambda}_i, \widehat{u}_i)\}_{i=1}^m$ for each $0 \le m \le k$, using $O(k^2(\mathcal{R}/\varepsilon)^4 \log(1/\delta))$ samples. If we set $\delta = \Theta(1/(|N_\lambda| \cdot |N_u|))^k$, then by a union bound the algorithm will successfully verify every candidate solution, and thus will succeed. Hence, we need to only draw $O(k^4(\mathcal{R}/\varepsilon)^4 \cdot \log(k\mathcal{R}/\varepsilon))$ samples.

This yields the claimed time/sample complexity bound of $(dk\mathcal{R}/\varepsilon)^{O(k^2)}$.

Remark 17 *As the above proof makes clear, the quadratic dependence on k in the $d^{O(k^2)}$ runtime for* NETLEARN *comes from the degree of the polynomial construction in Lemma 7, and the final brute force search (the latter only contributes to runtime, not to sample complexity). Note however that the upper bound of k^2 on the degree in this construction is somewhat pessimistic. Recall from the proof of Lemma 7 that the polynomial p defined in Eq. (13) has degree at most $|I| \cdot (k - |I|)$, where I is any one of the clusters of neurons indexed by $j \in J_{\mathsf{big}}$. In particular, if each such I is of constant size, e.g., then the degree of p is actually $O(k)$, and the sample complexity of* NETLEARN *(and the runtime barring the final brute force search) improves to $d^{O(k)}$.*

One simple situation where this happens is if all the weight vectors are $\Delta' = \mathrm{poly}(\varepsilon/\mathcal{R}, 1/k, 1/d)$-separated, in which case we can tune the Δ parameter in the analysis appropriately to ensure that all of the relevant intervals I are of size 1. For example, in the hard instance in the CSQ lower bound of Diakonikolas et al. (2020b), the weight vectors are of the form $u_j = \cos(\pi j/k) \cdot v + \sin(\pi j/k) \cdot w$ for orthogonal unit vectors v, w (see Eq. (3) therein) and are thus $\Omega(1/k)$-separated. Moreover, the span of $\{u_j\}$ has rank 2, and the proof of Lemma 13 implies the dimension of V is at most 2 as well. Thus, the runtime of brute force can also be reduced to exponential in k, rather than in k^2. Our algorithm is a CSQ algorithm,[2] and on this instance it has runtime matching the lower bound.

2. Note that we need to estimate $\mathbb{E}[(y - \widehat{f}(x))^2]$ for various choices of $\widehat{f}$ in Line 1 of NETLEARN, and technically this is not a CSQ. We can nevertheless remedy this by instead estimating $\mathbb{E}[2y \cdot \widehat{f}(x) - \widehat{f}(x)^2]$ and outputting the estimator $\widehat{f}$ which maximizes this quantity.

Acknowledgments

S.C. is supported by NSF Award #2103300. S.N. is supported by an NSF Graduate Fellowship and a Google Fellowship.

References

Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in neural information processing systems*, pages 6158–6169, 2019.

Ainesh Bakshi, Rajesh Jayaram, and David P Woodruff. Learning two layer rectified neural networks in polynomial time. In *Conference on Learning Theory*, pages 195–268. PMLR, 2019.

Sitan Chen, Adam R Klivans, and Raghu Meka. Efficiently learning any one hidden layer relu network from queries. *arXiv preprint arXiv:2111.04727*, 2021.

Sitan Chen, Adam R Klivans, and Raghu Meka. Learning deep relu networks is fixed-parameter tractable. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 696–707. IEEE, 2022.

Sitan Chen, Zehao Dou, Surbhi Goel, Adam R Klivans, and Raghu Meka. Learning narrow one-hidden-layer relu networks. *arXiv preprint arXiv:2304.10524*, 2023.

Amit Daniely. Sgd learns the conjugate kernel class of the network. *CoRR*, abs/1702.08503, 2017.

Ilias Diakonikolas and Daniel M Kane. Small covers for near-zero sets of polynomials and learning latent variable models. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 184–195. IEEE, 2020.

Ilias Diakonikolas and Daniel M. Kane. Efficiently learning one-hidden-layer relu networks via schur polynomials. *CoRR*, abs/2307.12840, 2023.

Ilias Diakonikolas, Surbhi Goel, Sushrut Karmalkar, Adam R Klivans, and Mahdi Soltanolkotabi. Approximation schemes for relu regression. In *Conference on Learning Theory*, 2020a.

Ilias Diakonikolas, Daniel M Kane, Vasilis Kontonis, and Nikos Zarifis. Algorithms and sq lower bounds for pac learning one-hidden-layer relu networks. In *Conference on Learning Theory*, pages 1514–1539, 2020b.

Weihao Gao, Ashok Vardhan Makkuva, Sewoong Oh, and Pramod Viswanath. Learning one-hidden-layer neural networks under general input distributions. *CoRR*, abs/1810.04133, 2018.

Rong Ge, Rohith Kuditipudi, Zhize Li, and Xiang Wang. Learning two-layer neural networks with symmetric inputs. In *International Conference on Learning Representations*, 2018a.

Rong Ge, Jason D Lee, and Tengyu Ma. Learning one-hidden-layer neural networks with landscape design. In *6th International Conference on Learning Representations, ICLR 2018*, 2018b.

Surbhi Goel and Adam R Klivans. Learning neural networks with two nonlinear layers in polynomial time. In *Conference on Learning Theory*, pages 1470–1499, 2019.

Surbhi Goel, Varun Kanade, Adam Klivans, and Justin Thaler. Reliably learning the relu in polynomial time. In *Conference on Learning Theory*, pages 1004–1042. PMLR, 2017.

Surbhi Goel, Adam R. Klivans, and Raghu Meka. Learning one convolutional layer with overlapping patches. In *ICML*, volume 80, pages 1778–1786. PMLR, 2018.

Surbhi Goel, Aravind Gollakota, Zhihan Jin, Sushrut Karmalkar, and Adam Klivans. Superpolynomial lower bounds for learning one-layer neural networks using gradient descent. In *International Conference on Machine Learning*, pages 3587–3596. PMLR, 2020.

Majid Janzamin, Hanie Sedghi, and Anima Anandkumar. Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. *arXiv*, pages arXiv–1506, 2015.

Yuanzhi Li and Yang Yuan. Convergence analysis of two-layer neural networks with relu activation. In *Advances in Neural Information Processing Systems 30*, pages 597–607, 2017.

Yuanzhi Li, Tengyu Ma, and Hongyang R. Zhang. Learning over-parametrized two-layer neural networks beyond ntk. In *Conference on Learning Theory 2020*, volume 125, pages 2613–2682. PMLR, 2020.

Hanie Sedghi, Majid Janzamin, and Anima Anandkumar. Provable tensor methods for learning mixtures of generalized linear models. In *Artificial Intelligence and Statistics*, pages 1223–1231. PMLR, 2016.

Mahdi Soltanolkotabi. Learning relus via gradient descent. In *Advances in neural information processing systems*, pages 2007–2017, 2017.

Santosh Vempala and John Wilmes. Gradient descent for one-hidden-layer neural networks: Polynomial convergence and sq lower bounds. In *COLT*, volume 99, 2019.

Qiuyi Zhang, Rina Panigrahy, and Sushant Sachdeva. Electron-proton dynamics in deep learning. *CoRR*, abs/1702.00458, 2017. URL http://arxiv.org/abs/1702.00458.

Xiao Zhang, Yaodong Yu, Lingxiao Wang, and Quanquan Gu. Learning one-hidden-layer relu networks via gradient descent. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1524–1534. PMLR, 2019.

Yuchen Zhang, Jason D Lee, and Michael I Jordan. L1-regularized neural networks are improperly learnable in polynomial time. In *33rd International Conference on Machine Learning, ICML 2016*, pages 1555–1563, 2016.

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