Learning Narrow One-Hidden-Layer ReLU Networks

Sitan Chen* UC Berkeley
Zehao Dou† Yale
Surbhi Goel‡ UPenn
Adam Klivans§ UT Austin
Raghu Meka¶ UCLA

April 21, 2023

Abstract

We consider the well-studied problem of learning a linear combination of k ReLU activations with respect to a Gaussian distribution on inputs in d dimensions. We give the first polynomial-time algorithm that succeeds whenever k is a constant. All prior polynomial-time learners require additional assumptions on the network, such as positive combining coefficients or the matrix of hidden weight vectors being well-conditioned.

Our approach is based on analyzing random contractions of higher-order moment tensors. We use a multi-scale analysis to argue that sufficiently close neurons can be collapsed together, sidestepping the conditioning issues present in prior work. This allows us to design an iterative procedure to discover individual neurons.

Contents

*Email: sitanc@berkeley.edu. Supported by NSF Award 2103300.
†Email: zehao.dou@yale.edu
‡Email: surbhig@cis.upenn.edu
§Email: klivans@cs.utexas.edu. Supported by NSF award AF-1909204 and the NSF AI Institute for Foundations of Machine Learning (IFML).
¶Email: raghum@cs.ucla.edu. Supported by NSF Tripods Institute grant 2217033.

1 Introduction

We study the problem of PAC learning one-hidden-layer ReLU networks from labeled examples. In particular, we consider ReLU networks with k neurons:

$$f_{\lambda,\mathbf{u}}(x) = \sum_{i=1}^{k} \lambda_i \cdot \mathsf{relu}(\langle u_i, x\rangle)$$

where $\mathsf{relu}(a) = \max(0, a)$ is the ReLU activation, and $u_1, \ldots, u_k \in \mathbb{S}^{d-1}$. Given examples of the form $(x, y) \in \mathbb{R}^d \times \mathbb{R}$ where x is drawn from a distribution $\mathcal{D}_{\mathcal{X}}$ and $y = f(x)$ for some $f \in \mathcal{F}$, our goal is to learn a function $\widehat{f} : \mathbb{R}^d \to \mathbb{R}$ with small test error, that is, $\mathbb{E}[(\widehat{f}(x) - y)^2] \le \varepsilon$.

This problem has inspired a large body of research in the machine learning community and acts as a benchmark for the design and analysis of novel learners. The goal is to design provably efficient (sample complexity and running time being polynomial in the problem parameters $d, k, 1/\varepsilon$) algorithms to PAC learn this class of functions under minimal assumptions. The most common assumption is that $\mathcal{D}_{\mathcal{X}}$ is the standard Gaussian distribution. Even under this assumption, no known algorithm achieves runtime and sample complexity $\mathrm{poly}(d, 1/\varepsilon)$ even for constant k.

This paper presents the first algorithm for PAC learning one-hidden-layer ReLU networks under Gaussian inputs that succeeds whenever k is constant:

Theorem 1.1. *Let $\mathcal{D}$ be the distribution over pairs $(x, y) \in \mathbb{R}^d \times \mathbb{R}$ where $x \sim \mathcal{N}(0, \mathrm{Id})$ and $y = f_{\lambda,\mathbf{u}}(x)$ for some $\lambda = (\lambda_1, \ldots, \lambda_k)$ and $\mathbf{u} = (u_1, \ldots, u_k)$. There is an algorithm that, given sample access to $\mathcal{D}$, has runtime and sample complexity $(d/\varepsilon)^{h(k)} \cdot \log(1/\delta)$ for $h(k) = k^{O(\log^2 k)}$ and outputs a function $\widehat{f}$ such that $\mathbb{E}[(\widehat{f}(x) - y)^2] \le \varepsilon$ with probability $1 - \delta$.*

As we explain in Remark 4.4, it is straightforward to verify that our algorithm also holds in the presence of unbiased i.i.d. noise on the labels y, so we omit these details for simplicity.

1.1 Technical overview

Tensors without a separation condition. The starting point for our approach is the standard fact that we can obtain estimates of *moment tensors* which encode high-order information about the unknown weight vectors u_i, namely tensors of the form $T_\ell \triangleq \sum_{i=1}^k \lambda_i u_i^{\otimes \ell}$. Indeed, given Gaussian examples $(x, f_{\lambda,\mathbf{u}}(x))$, we can estimate the expectation of $y \cdot S_\ell(x)$, where the ℓ-th order tensor-valued function S_ℓ is the *ℓ-th Hermite tensor*,[1] and recover approximate tensors $\widehat{T}_\ell$ such that $\|T_\ell - \widehat{T}_\ell\|_F \le \delta$ in time roughly $\ell^{O(\ell)} d^{2\ell}/\delta^2$.

At this juncture, many existing works in this literature (see Related Work) try to apply tensor decomposition on T_ℓ to recover the u_i's. Unfortunately, tensor decomposition is insufficient for us as it requires the weight vectors to be "non-degenerate" in some sense. This holds, for instance, if we assume u_i's are *well-separated*, meaning we have a non-negligible lower bound on $\|u_i - u_j\|_2$ for all $i, j \in [k]$ [MSS16]. Unfortunately, directly applying tensor decomposition will fail in the absence of such separation assumptions.

Clumping. To motivate our workaround, consider the simplest possible obstruction to the above approach: there exists a pair of indices $i \neq j$ such that $\|u_i - u_j\|_2$ is very small. The condition number of the weight vectors gets worse as this distance decreases, but intuitively if u_i, u_j are sufficiently close, we should be able to *clump* them together, that is, approximate them by a single neuron without incurring too much error in our approximation. While this seems promising, there is a critical hurdle here. In particular, suppose we group the k neurons into m clumps so that any two neurons in the same clump are distance at most γ from each other, and any two neurons in different clumps are distance at least γ from each other.[2] For every $i \in [m]$, let u'_i denote some representative vector from the clump, so that $\|u'_i - u'_j\| \ge \gamma$ for all $i \neq j$ as desired. We might hope to apply tensor decomposition to the tensor $T'_\ell = \sum_{j=1}^m \lambda_j (u'_j)^{\otimes \ell}$ to recover the representative vectors and thus learn the original network $f_{\lambda,\mathbf{u}}$. Unfortunately, there is a critical issue. We only have approximate access to T'_ℓ, but given the separation guarantee of γ on u'_i vectors, we would need an approximation to T'_ℓ with error $\delta' \ll \delta$; however, clumping vectors with distance δ will introduce error $\gg \delta$ in our tensor estimation. This quantitative trade-off will always be against us.

To get over the above, we have to introduce several new ideas. The core idea is to use a *multi-scale analysis* to pick which vectors to clump together strategically.

From tensors to random contractions. We will learn these clumps separately in multiple stages, rather than in "one shot" using tensor decomposition. To that end, instead of working with tensors T_ℓ, we will work with matrices

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

where $g \sim \mathcal{N}(0, I)$ is a random Gaussian vector. Given estimates for T_ℓ, we can form these by taking suitable tensor contractions. In place of tensor decomposition on T_ℓ, we will use PCA on M_ℓ^g for various ℓ. One challenge is that because we make no assumptions on $\lambda_1, \ldots, \lambda_k$, e.g. we do not assume they are nonnegative as in some prior works [DKKZ20, DK20, GLM18], many of these M_ℓ^g could be identically zero for all $g \in \mathbb{R}^d$, in which case PCA on such matrices provides no

[1] See Appendix A.1 for relevant background on Hermite polynomials.

[2] The careful reader will note that actually we can only ensure that neurons within the same clump are $k\gamma$-close and neurons within different clumps are γ-far, e.g. if the neurons lie on a line, but the extra factor of k isn't important to the present discussion.

information. In fact, [DKKZ20] gave a construction for which this is the case for all $\ell \le O(k)$ (see also [GGJ+20]). An important component of our analysis will be to argue that if we consider all ℓ up to a sufficiently large constant multiple of k, there actually is enough information across the different matrices M_ℓ^g to learn $f_{\lambda,\mathbf{u}}$.

First attempt: a single-stage algorithm. Let $v_i = \langle u_i, g\rangle$. It is not hard to see that, up to some $\mathrm{poly}(k,d)$ factors, $|v_i - v_j| \propto \|u_i - u_j\|_2$ with high probability, i.e. the amount of separation among the u_i's is inhereted by their projections v_i.

We can then do a case analysis. If $\max_{i\in[k]} v_i - \min_{i\in[k]} v_i \le \varepsilon'$ (for a suitable $\varepsilon = \varepsilon/\mathrm{poly}(d,k)$), then we can find an approximation to $f_{\lambda,\mathbf{u}}$ using just one neuron. On the other hand, suppose two of the v_i's are ε' far-away. For $i \in [k]$ and *scale* $\gamma > 0$,

$$S_i^{\mathsf{far}}(\gamma) \triangleq \{j \in [k] : |v_j - v_i| \ge \gamma\}$$
$$S_i^{\mathsf{close}}(\gamma) \triangleq \{j \in [k] : |v_j - v_i| \le T(\gamma)\},$$

where $T(\gamma) \approx (\gamma/d)^{O(k)}$ is a suitable parameter. For $\gamma \le \varepsilon'$, $S_i^{\mathsf{far}}(\gamma)$ will be nonempty, and $S_i^{\mathsf{close}}(\gamma)$ will consist only of v_j which are very close to v_i.

An easy case for us would be when every index i is *gapped* in the sense that v_j's for $j \ne i$ are either very close to v_i or very far from v_i, with nothing in between. Quantitatively, suppose there were a choice of $\gamma \le \varepsilon'$ such that $[k] = S_i^{\mathsf{close}}(\gamma) \sqcup S_i^{\mathsf{far}}(\gamma)$ for all $i \in [k]$. Then we could form clumps of neurons so that within any clump, any two neurons are $kT(\gamma)$-close, and any neurons in different clumps are γ-far. $kT(\gamma)$ is far smaller than γ, so that a certain PCA-based algorithm that works in the well-separated case can also be used to solve this gapped case. Furthermore, one can show that there always exists γ which is at least some value $\underline{\gamma}$ depending solely on the problem parameters (e.g. d, k, ε rather than the weight vectors themselves) for which we are in the gapped case. The issue is that the largest $\underline{\gamma}$ for which one can show this is of order $d^{-k^{\Theta(k)}}$, and this turns out to be tight – imagine $u_1, \dots, u_k$ lie on a line, and their pairwise separations scale roughly as $\varepsilon', T(\varepsilon'), T(T(\varepsilon')), \cdots$. Nevertheless, this strategy already gives a polynomial-time algorithm in the case of $k = O(1)$, with runtime $d^{k^{O(k)}}$. We give additional details for this approach in Section 3.

Better k dependence: a multi-stage algorithm. The bulk of this work is centered around refining the above guarantee with a multi-stage algorithm and multi-scale analysis to get a better dependence on k. The general idea is that it is not necessary to have a *single* scale under which every index i is simultaneously gapped. If we just want to learn a particular neuron i, we show that it suffices for there to be a scale γ under which i is gapped, even if no other indices are gapped at that scale (see Section 4.4). The proof of this relies on a certain estimate for power sum symmetric polynomials that may be of independent interest (Lemma 4.15). The key point is that if we just want γ under which at least one single neuron is gapped, it suffices to go down to scale of order $d^{-k^{\Theta(\log k)}}$, rather than $d^{-k^{\Theta(k)}}$, before such a γ exists (Lemma 4.6)

Our final algorithm then proceeds in stages. In each stage, either all of the remaining v_i's are ε'-close to each other, in which case we can approximate the network by a single neuron. Otherwise, we identify a set of indices i, each of which has a corresponding scale γ under which it is gapped, and argue that the set of top k principal components across all M_ℓ^g with $\ell \le O(k)$ spans a subspace containing i. By enumerating over this subspace, we can learn the gapped neurons and make progress. We then recurse on the residual network given by subtracting the contribution from the neurons we have learned.

There is one last subtlety: given an approximation to the residual network at any given step of this algorithm, if the approximation error is ξ, then it turns out this error gets blown up,

Algorithm 1: MULTISCALELEARN(f)

Sample random unit vector $g \in \mathbb{S}^{d-1}$
$\widehat{\lambda} \leftarrow \emptyset$, $\widehat{\mathbf{u}} \leftarrow \emptyset$
for $t = 1, \dots, k$ **do**
  for $\ell = 1, 2, 4, \dots, 2k+2$ **do**
    Compute estimates $\widehat{T}_\ell$ of $\mathbb{E}[(y - f_{\widehat{\lambda},\widehat{\mathbf{u}}}(x)) \cdot S_\ell(x)]$ from samples
    Evaluate $\widehat{M}_\ell \leftarrow \widehat{T}_\ell(g, \cdots, g, :, :)$
  Form a candidate estimate h for the residual $f - f_{\widehat{\lambda},\widehat{\mathbf{u}}}$ as a neural network of the form $\mu^+\mathsf{relu}(\langle u, \cdot\rangle) + \mu^-\mathsf{relu}(\langle -u, \cdot\rangle)$ (see proof of Lemma 4.5)
  Compute the top-k singular subspaces of $\widehat{M}_2, \dots, \widehat{M}_{2k+2}$ and let V be the joint $O(k^2)$-dimensional span of these subspaces.
  Form nets over V and over $[-\mathcal{R}, \mathcal{R}]$ of granularity roughly $\mathrm{poly}(d, 1/\varepsilon, \mathcal{R})^{-k^{\Omega(\log^2 k)}}$ and guess an integer $m \in \{1, \dots, k\}$ and elements $u'_1, \dots, u'_m$ and $\lambda'_1, \dots, u'_m$ from each of these nets.
  (Nondeterministically) either add μ^+, μ^- and $u, -u$ to $\widehat{\lambda}$ and $\widehat{\mathbf{u}}$, or add $\lambda'_1, \dots, \lambda'_m$ and $u'_1, \dots, u'_m$ to $\widehat{\lambda}$ and $\widehat{\mathbf{u}}$, or break out of the loop.
  Estimate $\|f - f_{\widehat{\lambda},\widehat{\mathbf{u}}}\|^2$ from samples. If this is small, terminate and **return** $f_{\widehat{\lambda},\widehat{\mathbf{u}}}$.
return *Fail*

roughly speaking, to $\xi^{1/\Theta(k^{\log k})}$ in the next step of the algorithm. As a result, in order for the approximation error to still be small after T iterations, we must estimate the matrices M_ℓ^g to error scaling exponentially in $k^{T \log k}$. Naively one can only ensure that a single new neuron is learned in each stage of the algorithm, meaning T could potentially be as large as k, in which case we obtain no improvement over the single-stage algorithm above. Instead, via a careful combinatorial argument (Section 5), we show that it is possible to learn enough neurons in each stage of the algorithm that we terminate in $T \le O(\log k)$ stages (Lemma 4.13), thus yielding the improved runtime of $d^{k^{\log^2 k}}$ claimed in Theorem 1.1.

The above procedure is summarized in the pseudocode for our algorithm (see Algorithm 1). We present it as a nondeterministic algorithm, but there are only $(d/\varepsilon)^{k^{O(\log^2 k)}}$ possible choices in each iteration of the loop, for a total of $(d/\varepsilon)^{k^{O(\log^2 k)}}$ computation paths. To form our final estimator, we simply try each of these paths, and as our rigorous guarantees imply that one of these paths yields a valid estimator, we output the $f_{\widehat{\lambda},\widehat{\mathbf{u}}}$ which achieves the best empirical loss.

1.2 Related work

Algorithms for PAC learning neural networks. The design and analysis of algorithms for PAC learning various classes of simple neural networks has been very active in the last several years and has led to many innovative algorithms. These works make assumptions on the distribution of the inputs, the noise in the label, and the structure of the neural network to sidestep a large body of computational hardness results [SSSS17, MR18, Sha18, VW19, DV20, GGJ+20, DKKZ20].

Examples of algorithmic techniques involved include tensor decomposition [JSA15, SJA16, BJW19, GLM18, GKLW18, GMOV18, DKKZ20], kernel methods [ZLJ16, GKKT17, Dan17, GK19], trajectory analyses of gradient-based methods [ZSJ+17, LY17, VW19, ZYWG19, Sol17, ZPS17,

DGK+20, LMZ20, GKM18, AZLL19], filtered PCA [CKM22], and explicit covers for algebraic varieties [DK20].

Despite this flurry of work, the complexity of learning one-hidden-layer ReLU networks with respect to Gaussians remains open. As mentioned above, the only results that achieve runtime and sample complexity polynomial in d, k and $1/\varepsilon$ require additional assumptions on the structure of the network, in particular (i) the matrix constructed from the weight parameters in the network is well-conditioned and/or (ii) the output layer weights are all positive. Under assumption (i), parameter recovery becomes possible, which is sufficient but unnecessary for PAC learning. The only known results that do not require a condition number bound (and hence do not recover parameters) are by [DKKZ20] and [CKM22]. The former requires assumption (ii), while the latter pays an exponential dependence on $1/\varepsilon$ even for constant k. Our result removes assumptions (i) and (ii) and gets a polynomial dependence in the error parameter for constant k.

Statistical query algorithms. Recent results by [GGJ+20] and [DKKZ20] rule out a $d^{o(k)}$ time algorithm for PAC learning one-hidden-layer ReLU networks for a large class of algorithms (including gradient descent on square loss) known as *correlational statistical query* (CSQ) algorithms. A CSQ algorithm is allowed to access the data only via noisy correlational queries of the form $\mathbb{E}[yf(x)]$ for any query f chosen by the learner. Our algorithm fits into this paradigm of CSQ algorithms and hence suffers from a $d^{\Omega(k)}$ runtime. Before our result, no known CSQ algorithm achieved $d^{r(k)}$ for any function r, which is independent of d without additional assumptions on the structure of the network. We note that the recent result by [CKM22] that achieves a polynomial dependence on d for general networks is not a CSQ algorithm.

2 Technical Preliminaries

Notation. Given $f \in L^2(\mathbb{R}^d, \omega_d)$, where ω_d is the standard Gaussian measure on $\mathbb{R}^d$, let $\|f\|_2$ denote its L_2 norm, that is, $\|f\|_2^2 = \mathbb{E}_{x\sim\mathcal{N}(0,\mathrm{Id})}[f(x)^2]$.

2.1 ReLU networks and moment tensors

It will be convenient to express one-hidden-layer ReLU networks in the following form, as the sum of a linear function with a linear combination of absolute values:

Lemma 2.1. *Given a one-hidden-layer ReLU network $f(x) = \sum_{i=1}^k \mu_i \cdot \mathsf{relu}(\langle u_i, x\rangle)$, there exist $w \in \mathbb{R}^d$ and $\lambda_1, \ldots, \lambda_k \in \mathbb{R}$ such that*

$$f(x) = \langle w, x\rangle + \sum_{i=1}^k \lambda_i \cdot |\langle u_i, x\rangle|$$

for all $x \in \mathbb{R}^d$. Furthermore, $\|w\| \le \sum_i |\lambda_i|$.

Proof. Note that $\mathsf{relu}(z) = |z|/2 + z/2$ for any $z \in \mathbb{R}$, so we can write

$$f(x) = \sum_{i=1}^k \frac{\mu_i}{2} \cdot |\langle u_i, x\rangle| + \left\langle \sum_{i=1}^k \frac{\mu_i}{2} \cdot u_i, x \right\rangle.$$

We can thus take $w \triangleq \sum_i \frac{\mu_i}{2} \cdot u_i$ and $\lambda_i \triangleq \mu_i/2$. The last part of the lemma is immediate. □

In light of Lemma 2.1, 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

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

When the λ_i, u_i, and k are clear from context, we denote $\{\lambda_i, u_i\}_{i\in[k]}$ by $(\lambda, \mathbf{u})$. Given $S \subseteq [k]$, we denote $\{\lambda_i, u_i\}_{i\in S}$ by $(\lambda_S, \mathbf{u}_S)$. We call k the *width* of the network $f_{w,\lambda,\mathbf{u}}$. Our bounds will depend on the L_1 norm of λ. Thus, henceforth suppose $\|\lambda\|_1 \le \mathcal{R}$ for some parameter $\mathcal{R} \ge 1$. Note that by the last part of Lemma 2.1, we have

$$\|w\| \le \|\lambda\|_1 \le \mathcal{R} \,.$$

Given $(\lambda_1, u_1), \dots, (\lambda_k, u_k) \in \mathbb{R} \times \mathbb{S}^{d-1}$, $g \in \mathbb{S}^{d-1}$, and $\ell \in \mathbb{N}$, define

$$T_\ell(\{\lambda_i, u_i\}_{i\in[k]}) \triangleq \sum_i \lambda_i u_i^{\otimes \ell} \qquad \text{and} \qquad M_\ell^g(\{\lambda_i, u_i\}_{i\in[k]}) \triangleq \sum_i \lambda_i \langle u_i, g\rangle^{\ell-2} \cdot u_i u_i^\top \,,$$

noting that M_ℓ^g can be obtained by contracting the *moment tensor* T_ℓ along the direction g for all of the first $\ell - 2$ modes. When g is clear from context, we denote M_ℓ^g by M_ℓ.

Given $(\lambda, \mathbf{u}), (\lambda', \mathbf{u}') \in (\mathbb{R} \times \mathbb{S}^{d-1})^k$, define the *parameter distance*

$$d_{\mathsf{param}}((\lambda, \mathbf{u}), (\lambda', \mathbf{u}')) \triangleq \min_\pi \max_i \{|\lambda_i - \lambda'_{\pi(i)}| + \|u_i - u'_{\pi(i)}\|\} \,,$$

where the minimization is over all possible permutations of k elements.

2.2 Anti-concentration

A main component of our algorithm is to contract estimates for the moment tensors T_ℓ along a random unit direction g to get the matrices M_ℓ^g defined in the previous section. The most important feature of this operation is that it roughly preserves distances in the sense that if two weight vectors u_i, u_j are close/far, their projections $\langle u_i, g\rangle$ and $\langle u_j, g\rangle$ are as well. Formally, we have the following elementary bounds, which follow by standard anti-concentration (see Appendix A.3).

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

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

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

Lemma 2.3. *With probability at least* $9/10$ *over random* $g \in \mathbb{S}^{d-1}$*, we have that* $|\langle u_i, g\rangle| \ge c/(k\sqrt{d})$ *for all* i*, for some absolute constant* $c > 0$*.*

Henceforth, we will condition on the event that g satisfies Lemmas 2.2 and 2.3. We will denote

$$v_i \triangleq \langle u_i, g\rangle \tag{1}$$

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

$$0 \le v_1 \le \cdots \le v_k \,.$$

3 First Attempt: Learning in Time $d^{k^{O(k)}}$

Let us recall the technique used by [DKKZ20] to PAC learn one-hidden-layer ReLU networks with non-negative combining weights λ. The approach is to approximately recover the subspace U spanned by $\mathbf{u}$ using the matrix of degree-2 Chow parameters (or the second moment matrix), that is, $\mathbb{E}[y \cdot S_2(x)] = \sum_{i=1}^k \lambda_i u_i u_i^\top$. From here, one can see that the span of this second moment matrix can exactly recover the subspace U spanned by $\mathbf{u}$. Using the non-negativity of λ, [DKKZ20] argue that the span of the eigenvectors corresponding to the top-k eigenvalues of an approximate second moment matrix (computed using samples) contains k-vectors $\hat{\mathbf{u}}$ such that $\|f_{\lambda,\mathbf{u}} - f_{\lambda,\hat{\mathbf{u}}}\|_2$ is small. In fact, they show a stronger property: for all $i \in [k]$, $\lambda_i \|u_i - \hat{u}_i\|^2$ is small. Following this, a brute force strategy on this subspace recovers an approximately-good hypothesis.

For general, possibly negative, combining weights λ, if we can design a matrix $M \approx \sum_{i=1}^k |\lambda_i| u_i u_i^\top$ which we can estimate using samples, then we can use the technique from [DKKZ20] to guarantee recovery of a k-dimensional subspace such that for all $i \in [k]$, $|\lambda_i| \|u_i - v_i\|^2$ is small, which will guarantee small loss. Our first idea is to take M to be a suitable linear combination of the moment tensor contractions $M_\ell^g = \sum_{i=1}^k \lambda_i \langle u_i, g\rangle^{\ell-2} \cdot u_i u_i^\top$. That is, we would like to find coefficients $\{\alpha_s\}_{s\in[k]}$ such that

$$\sum_{s=1}^k \alpha_s M_{2s}^g \approx \sum_{i=1}^k |\lambda_i| u_i u_i^\top .$$

As long as the entries of α are not too large in magnitude, we can use a net-argument to brute-force over the choices of α and run [DKKZ20] for each choice of α. Showing that there exists such α_s for $s \in [k]$ reduces to showing that for all $i \in [k]$

$$\sum_{s=1}^k \alpha_s v_i^{2(s-1)} = \operatorname{sgn}(\lambda_i).$$

This is equivalent to showing the existence of a univariate polynomial p of degree $k-1$ with bounded norm that matches the sign pattern of λ on inputs $\{v_1^2, \ldots, v_k^2\}$.

If we had that $|v_i^2 - v_j^2|$ was lower bounded for all $i \neq j$, then using the following condition number bound for the $k \times k$ Vandermonde matrix generated by $v_1^2, \ldots, v_k^2$ would give us the desired α with bounded norm dependent on the gap:

Lemma 3.1 (E.g., Fact 5.10 from [CLLZ22])**.** *Given an $m \times m$ Vandermonde matrix V generated by nodes $a_1, \ldots, a_m$ for which $|a_i - a_j| \geq \Delta$ for all $i \neq j$, and given $c \in \mathbb{R}^m$, there exists α for which $V\alpha = c$ such that $\|\alpha\| \leq m(1/\Delta)^{2m-2}\|c\|$.*

Unfortunately, as we make no assumptions on the hidden weight vectors, it is not necessarily the case that $v_1^2, \ldots, v_k^2$ are well-separated. Nevertheless, we could try clumping together very close v_i^2's into a single representative node (and adding their corresponding combining coefficients) so that each clump is well-separated while the approximation error from clumping does not blow up. In order to formally define clumping at scale γ, consider a graph on the indices with an edge between indices i, j if $|v_i^2 - v_j^2| \leq \gamma$. Then a clumping is specified by a set of disjoint connected subgraphs in this graph. The main challenge is that if we clump things together at scale γ, then when we use the above result to construct $\{\alpha_s\}$, our coefficients are of magnitude $O(1/\gamma^k)$. This would lead to an $O(1/\gamma^k)$ blow up in the error for the indices within each clump when we approximate them by the representative node for that clump.

If we can find a scale γ such that the elements in any clump are $\approx \gamma^k$-close while the clumps are γ-separated from each other, then this blow up does not hurt us. It turns out that we can always

find a scale $\gamma = \varepsilon^{k^c}$ for some $c \in [0, k-1]$ that satisfies this γ versus γ^k "gap." To prove this, suppose we are at a scale ε^{k^c} and this property is not satisfied. Then there must be two clumps that are separated by $< \varepsilon^{k^c}$, therefore, when we go up a scale $\varepsilon^{k^{c-1}}$ then these two clumps would be combined together. This implies that whenever our condition is not satisfied, going up a scale reduces the number of clumps by 1. However, there can be at most k clumps at the smallest scale. Thus at some scale, we must either have our desired gap or we can clump everything together. If we can clump everything together, this implies that the original network is well-approximated by at most two neurons, and we can learn these neurons directly.

This attempt would give us a runtime of $(d/\varepsilon)^{k^{O(k)}}$. This argument can be formalized, however we only present the high-level intuition since our main result improves over this significantly. At a high level, the improvement is as follows. So far, we have given an approach that tries to learn the network in "one shot" by looking for a single scale at which there is a gap for all clumps simultaneously. Instead, in our improved algorithm, we learn the network over multiple steps. In each step, we identify disjoint clumps at several different scales such that each clump has a gap *for its corresponding scale*, and use PCA to learn the neurons within these clumps. It turns out that instead of going down to scales as small as ε^{k^k}, now it suffices to go down to scale $\varepsilon^{k^{\log k}}$ (Lemma 4.6). We then prove that we can find enough such clumps at each step that after $O(\log k)$ steps, we can approximate the entire network, thus yielding an improved runtime of $(d/\varepsilon)^{k^{\log^2 k}}$.

4 Recursively Learning in Time $d^{k^{O(\log^2 k)}}$

As the algorithmic guarantee in Theorem 1.1 only beats brute force for $k \ll \mathrm{poly}(d)$, throughout we will assume this to simplify some estimates.

For some $S \subseteq [k]$, suppose we have access to $(\widehat{\lambda}_i, \widehat{u}_i)_{i \in S}$ for which

$$\|f_{w,\lambda,\mathbf{u}} - f_{0,\widehat{\lambda},\widehat{\mathbf{u}}} - f_{w,\lambda_{S^c},\mathbf{u}_{S^c}}\| \le \xi\,. \tag{2}$$

In other words, $f_{w,\lambda,\mathbf{u}} - f_{0,\widehat{\lambda},\widehat{\mathbf{u}}}$, i.e. the difference between the ground truth and what we have learned so far, is close to $\langle w, x\rangle$ plus the subnetwork of f indexed by S^c. We will refer to this latter network as

$$f_{\mathsf{res}} \triangleq f_{w,\lambda_{S^c},\mathbf{u}_{S^c}}\,.$$

For notational convenience, we assume without loss of generality that $S = \{k_{\mathsf{res}}+1, \dots, k\}$, where $k_{\mathsf{res}} \triangleq k - |S|$. Define the parameters

$$\varepsilon' \triangleq \varepsilon/\mathrm{poly}(d,k,\mathcal{R}) \qquad \Lambda \triangleq \mathrm{poly}(d\mathcal{R}/\varepsilon)\cdot \xi^{1/k^{\Theta(\log k)}} \qquad \underline{\gamma} \triangleq \left(\frac{\Lambda\varepsilon}{\mathcal{R}d}\right)^{k^{\Theta(\log k)}}. \tag{3}$$

The parameter ξ will be sufficiently small that $\Lambda \ll 1$.

One important case in which we will show it is possible to learn a neuron $i \in [k_{\mathsf{res}}]$ is when there is a significant *gap* among the distances from other v_j to v_i, i.e. every v_j is either very far from v_i or very close to v_i. Formally, given $i \in [k_{\mathsf{res}}]$ and $\gamma > 0$, define

$$S_i^{\mathsf{far}}(\gamma) \triangleq \{j \in [k_{\mathsf{res}}] : |v_j - v_i| \ge \gamma\}$$
$$S_i^{\mathsf{close}}(\gamma) \triangleq \{j \in [k_{\mathsf{res}}] : |v_j - v_i| \le T(\gamma)\}\,,$$

where

$$T(\gamma) \triangleq \frac{\Lambda^{10}}{\mathcal{R}^2}\,(\gamma/d)^{\Theta(k)}\,. \tag{4}$$

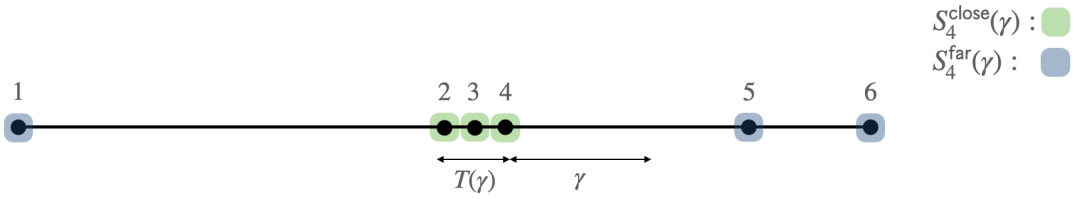

Figure 1: Illustration of $S_i^{\mathsf{close}}(\gamma)$ and $S_i^{\mathsf{far}}(\gamma)$. The figure shows that γ is a gapped scale for $i = 4$, but note that because points 5 and 6 are distance strictly less than γ apart and strictly greater than $T(\gamma)$ apart, γ is not a gapped scale for $i = 5, 6$.

Definition 4.1. *Given $i \in [k_{\mathsf{res}}]$ and $0 < \gamma < 1$, we say that γ is a* gapped scale for i *if $[k_{\mathsf{res}}] = S_i^{\mathsf{close}}(\gamma) \sqcup S_i^{\mathsf{far}}(\gamma)$ and $\gamma \geq \underline{\gamma}$. Further, we say that i is* detectable at scale γ *if $|\sum_{j \in S_i^{\mathsf{close}}(\gamma)} \lambda_i| > \Lambda$.*

In this section we will show that one of two things can happen which will allow us to learn part of or all of f_{res}.

1. If $\max_{i \in [k_{\mathsf{res}}]} v_i - \min_{i \in [k_{\mathsf{res}}]} v_i \leq \varepsilon'$ (this includes the case where $k_{\mathsf{res}} = 0$), then we can find an approximation to f_{res} as a linear combination of two ReLUs and terminate.

2. There exists at least one pair $(i, \gamma) \in [k_{\mathsf{res}}] \times [\underline{\gamma}, 1)$ such that γ is a gapped scale for i. In this case, let $\mathcal{J}$ be any set of such pairs (i, γ) such that the sets $S_i^{\mathsf{close}}(\gamma)$ are all disjoint (we will specify the precise $\mathcal{J}$ that we work with later). We will show that:

   (a) For those $(i, \gamma) \in \mathcal{J}$ for which i is detectable at scale γ, we can use the different M_ℓ to construct a net containing a vector close to u_j for some $j \in S_i^{\mathsf{close}}(\gamma)$.

   (b) For all other $(i, \gamma) \in \mathcal{J}$, the subnetwork f'_{res} of f_{res} given by the corresponding neurons in f_{res} is well-approximated by the zero function.

   Suppose we know which $(i, \gamma) \in \mathcal{J}$ fall into Case 2a versus Case 2b (recall that our algorithm is nondeterministic, so along some computation path we will have correctly guessed these). Then after brute-forcing over weights $\widehat{\lambda}$ to assign to the neurons learned in Case 2a, we can update the set of pairs $(\widehat{\lambda}_i, \widehat{u}_i)_{i \in S}$ to a set of pairs $(\widetilde{\lambda}_i, \widetilde{u}_i)_{i \in S'}$ for some $S' \supsetneq S$ such that

$$\|f_{w,\lambda,\mathbf{u}} - f_{0,\widetilde{\lambda},\widetilde{\mathbf{u}}} - f_{w,\lambda_{S'^c},\mathbf{u}_{S'^c}}\| \leq \xi^*$$

   for some new error ξ^*. We can then recurse, until the set S' is all of $[k]$ and we have learned an approximation to the entire original network $f_{w,\lambda,\mathbf{u}}$.

A priori, one might need up to k recursive steps to learn $f_{w,\lambda,\mathbf{u}}$, but across this many steps the initial error ξ would be blown up by a factor which is doubly exponential in k. We will instead argue that starting with $S = \emptyset$, there is sequence of choices of $\mathcal{J}$ across only $O(\log k)$ recursive steps such that we end up terminating with a sufficiently good approximation to the original network $f_{w,\lambda,\mathbf{u}}$. This will require a delicate combinatorial argument (Section 5) which is crucial to obtaining our $d^{\mathsf{quasipoly}(k)}$ runtime and sample complexity.

4.1 Moment tensor estimations

Here we show how to estimate $T_\ell(\lambda_{[k_{\mathsf{res}}]}, \mathbf{u}_{[k_{\mathsf{res}}]})$ using the parameters $(\widehat{\lambda}, \widehat{\mathbf{u}})$ as well as Gaussian samples $\{(x_a, y_a)\}_{a \in [N]}$ labeled by the original function $f_{w,\lambda,\mathbf{u}}$. We use the following standard

guarantee for approximately recovering moment tensors from samples, whose proof is deferred to Appendix B.1.

Lemma 4.2 (Moment estimation)**.** *Let $\ell \in \{1, 2, 4, 6, \ldots\}$. Let*

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

Given samples $\{(x_i, f_{w,\lambda,\mathbf{u}}(x_i)\}_{i=1,\ldots,N}$ for $x_i \sim \mathcal{N}(0, \mathrm{Id})$ and $N \geq \ell^{O(\ell)} d^{2\ell} \mathcal{R}^2/\xi^2$, the tensor $\widehat{T} = \frac{1}{2c_\ell N} \sum_{i=1}^k f_{w,\lambda,\mathbf{u}}(x_i) \cdot S_\ell(x_i)$, where S_ℓ is the ℓ-th normalized probabilist's Hermite tensor, satisfies $\|\widehat{T} - T_\ell(\lambda, \mathbf{u})\|_F \leq \xi$ if ℓ is even, and otherwise satisfies $\|\widehat{T} - w\|_2 \leq \xi$ if $\ell = 1$.

The following lemma, whose proof is deferred to Appendix B.2, shows that we can approximate the Hermite coefficients of f_{res} by empirically estimating the Hermite coefficients of $f_{w,\lambda,\mathbf{u}} - f_{0,\widehat{\lambda},\widehat{\mathbf{u}}}$:

Lemma 4.3. *Let*

$$\xi' \triangleq \mathrm{poly}(d) \cdot (\mathcal{R} + \|\widehat{\lambda}\|_1) \cdot \xi \,.$$

Given the parameters $(\widehat{\lambda}, \widehat{\mathbf{u}})$ and Gaussian samples $(x_1, y_1), \ldots, (x_N, y_N)$ labeled by the original function $f_{w,\lambda,\mathbf{u}}$, we have for all $\ell = 2, \ldots, 2k+2$ that

$$\left\| \frac{1}{N} \sum_{a=1}^N (y_a - f_{0,\widehat{\lambda},\widehat{\mathbf{u}}}(x_a))\, x_a - w \right\| \leq \xi'$$

$$\left\| \frac{1}{2c_\ell N} \sum_{a=1}^N (y_a - f_{0,\widehat{\lambda},\widehat{\mathbf{u}}}(x_a))\, S_\ell(x_a) - T_\ell(\lambda_{[k_{\mathsf{res}}]}, \mathbf{u}_{[k_{\mathsf{res}}]}) \right\| \leq \xi'$$

provided $N \geq d^{O(k)}\, (\mathcal{R}^2 + \|\widehat{\lambda}\|_1^2)/\xi'^2$.

Remark 4.4. *Note that if instead of getting samples of the form $(x, f_{w,\lambda,\mathbf{u}}(x))$, we had samples of the form $(x, f_{w,\lambda,\mathbf{u}}(x) + \zeta)$ where ζ is independent mean-zero noise, then the estimators for $T_\ell(\lambda, \mathbf{u})$ and w defined above are still unbiased estimators for these quantities. Our algorithm uses its samples solely to form these estimators, so provided ζ has bounded second moment so that the variance of these estimators is not too large, our algorithm will still work in the presence of such label noise.*

4.2 Case 1: two-neuron approximation

Lemma 4.5. *Suppose $\max_{i \in [k_{\mathsf{res}}]} v_i - \min_{i \in [k_{\mathsf{res}}]} v_i \leq \varepsilon'$. Then there is an efficient algorithm that takes the parameters $(\widehat{\lambda}, \widehat{\mathbf{u}})$ as well as $\mathrm{poly}(d)(\mathcal{R}^2 + \|\widehat{\lambda}\|_1^2)/\xi'^2$ samples from the original function $f_{w,\lambda,\mathbf{u}}$ and outputs weights μ^+, μ^- and a vector u such that the network*

$$h \triangleq \mu^+ \mathsf{relu}(\langle u, \cdot \rangle) + \mu^- \mathsf{relu}(\langle -u, \cdot \rangle)$$

satisfies $\|f_{w,\lambda,\mathbf{u}} - f_{0,\widehat{\lambda},\widehat{\mathbf{u}}} - h\| \lesssim \mathrm{poly}(d, \mathcal{R})(\varepsilon' + \xi')$.

Sketch, see Appendix B.3. The condition on $v_1, \ldots, v_{k_{\mathsf{res}}}$ and the fact that projection along g roughly preserves distances among $u_1, \ldots, u_{k_{\mathsf{res}}}$ implies that all of the weight vectors in f_{res} are close in Euclidean distance. As a result, f_{res} is well-approximated by a ReLU network which only depends on the projection of the input to a single direction, i.e. a width-2 network whose weight vectors are u and $-u$ for some vector u. This can be done by considering suitable linear combinations of the first Hermite coefficients of f_{res}, namely w, together with a contraction of the second moment matrix, which is simply $\sum_{i \in [k_{\mathsf{res}}]} \lambda_i v_i u_i$, both of which can be estimated from samples by Lemma 4.3. $\square$

4.3 Existence of a gapped scale

As a warmup, here we show that if we are not in Case 1, there exists $i \in [k_{\mathsf{res}}]$ for which there is a gapped scale γ.

Lemma 4.6. *Suppose* $\max_{i\in[k_{\mathsf{res}}]} v_i - \min_{i\in[k_{\mathsf{res}}]} v_i > \varepsilon'$. *Then there exists at least one index* $i \in [k_{\mathsf{res}}]$ *for which there is a gapped scale* γ.

Proof. In this proof, assume without loss of generality that $v_1 \le \cdots \le v_{k_{\mathsf{res}}}$, and define $\delta_i \triangleq v_i - v_{i-1}$ for $1 < i \le k_{\mathsf{res}}$. By averaging, there exists some i_0 for which $\delta_{i_0} \ge \varepsilon'/k_{\mathsf{res}}$. Suppose without loss of generality that $i_0 > k_{\mathsf{res}}/2$.

If $\delta_j \le T(\varepsilon'/k_{\mathsf{res}})/k_{\mathsf{res}}$ for all $i_0 < j \le k_{\mathsf{res}}$, then $\varepsilon'/k_{\mathsf{res}}$ is a gapped scale for i_0. Otherwise, there exists $i_0 < i_1 \le k_{\mathsf{res}}$ for which $\delta_{i_1} > T(\varepsilon'/k_{\mathsf{res}})/k_{\mathsf{res}}$. Suppose without loss of generality that i_1 is closer to k_{res} than i_0.

Again, if $\delta_j \le T(T(\varepsilon'/k_{\mathsf{res}})/k_{\mathsf{res}})/k_{\mathsf{res}}$ for all $i_1 < j \le k_{\mathsf{res}}$, then $T(\varepsilon'/k_{\mathsf{res}})/k_{\mathsf{res}}$ is a gapped scale for i_0. Otherwise, we can continue this binary search procedure at most $O(\log k)$ times.

Finally, we verify that $\underline{\gamma}$ is smaller than the result of iterating $z \mapsto T(z/k_{\mathsf{res}})/k_{\mathsf{res}}$ for $O(\log k)$ times starting from ε', which will prove that there exists a gapped scale γ for some $i \in [k_{\mathsf{res}}]$.

To verify this, note that $T(z/k_{\mathsf{res}})/k_{\mathsf{res}} \ge T(z/k)/k \ge (\Lambda^{10}/\mathcal{R}^2)(z/d)^{\Theta(k)}$. So iterating this $O(\log k)$ times starting from $z = \varepsilon'$ gives a quantity which is at least $\left(\frac{\Lambda\varepsilon}{\mathcal{R}d}\right)^{k^{\Theta(\log k)}} \approx \underline{\gamma}$. □

4.4 Case 2a: detectable neurons

Here we show that for any (i, γ) for which γ is a gapped scale for i and furthermore i is detectable, we can use a certain PCA-based procedure to produce a net over vectors, at least one of which is close to u_i.

The main ingredient in the proof is the following consequence of an estimate for power sums (Lemma 4.15) that we prove in Section 4.7.

Lemma 4.7. *Consider* $i \in [k_{\mathsf{res}}]$ *and* γ *which is a gapped scale for* i *and such that* i *is detectable at* γ. *Then for any projector* $\Pi \in \mathbb{R}^{d\times d}$ *and* $r \triangleq \Pi u_i$, *we have that*

$$r^\top M_\ell(\lambda_{[k_{\mathsf{res}}]}, \mathbf{u}_{[k_{\mathsf{res}}]})\, r \ge C_1 \|r\|_2^4 - C_2$$

for

$$C_1 = \Lambda\,(\gamma/d)^{\Theta(k)} \qquad C_2 \lesssim \mathcal{R}\mathrm{poly}(d)\, T(\gamma)^{1/2}\,.$$

for some even integer $2 \le \ell \le 2k+2$ *and absolute constants* $c, C > 0$.

Proof. Take k and k' in Lemma 4.15 to be k_{res} and $|S_i^{\mathsf{close}}(\gamma)|$ respectively. Permute $[k_{\mathsf{res}}]$ so that i is now the first element, $S_i^{\mathsf{close}}(\gamma)$ consists of the first k' elements, and $S_i^{\mathsf{far}}(\gamma)$ consists of the remaining. For every $i \in [k_{\mathsf{res}}]$, let $q_i = \lambda_i \langle u_i, r\rangle^2$, and let v_i be as defined in (1). Note that

$$r^\top M_\ell(\lambda_{[k_{\mathsf{res}}]}, \mathbf{u}_{[k_{\mathsf{res}}]})\, r = \sum_{i=1}^{k_{\mathsf{res}}} \lambda_i \langle u_i, g\rangle^{\ell-2}\langle u_i, r\rangle^2 = \sum_{i=1}^{k_{\mathsf{res}}} q_i v_i^\ell = \langle v^{\odot\ell}, q\rangle\,. \tag{5}$$

As we are conditioning throughout on the event of Lemma 2.3, we have that $|v_i| \ge \alpha$ for all i for $\alpha \triangleq c/(k\sqrt{d})$. We can take R in Lemma 4.15 to be the assumed bound $\mathcal{R}$ on $\|\lambda\|_1$, as $\|q\|_\infty \le \|\lambda\|_\infty \le \mathcal{R}$. And by the hypothesis of the lemma, $|v_i - v_j| \le \beta$ for all $j \in S_i^{\mathsf{close}}(\gamma)$ for $\beta \triangleq T(\gamma)$ and $|v_i - v_j| \ge \gamma$ for all $j \in S_i^{\mathsf{far}}(\gamma)$.

By Lemma 4.15 and (5), we conclude that

$$\begin{aligned} r^\top M_\ell(\lambda_{[k_{\mathsf{res}}]}, \mathbf{u}_{[k_{\mathsf{res}}]})\, r &\geq \frac{\tau}{2k_{\mathsf{res}}}\Big(\frac{c^2\gamma^2}{4k^3 d}\Big)^{k_{\mathsf{res}}} - C\mathcal{R}k_{\mathsf{res}}\left(|S_i^{\mathsf{close}}(\gamma)| - 1\right) T(\gamma) \\ &\geq \frac{\tau}{2k}\Big(\frac{c^2\gamma^2}{4k^3 d}\Big)^{k} - C\mathcal{R}k^2 T(\gamma) \end{aligned}$$

for $\tau \triangleq |\sum_{j\in S_i^{\mathsf{close}}(\gamma)} q_j|$. It remains to relate τ to $\|r\|^4$. Recalling that $r = \Pi u_i$, we have

$$\langle u_i, r\rangle^2 = \langle u_i, \Pi u_i\rangle^2 = \|\Pi u_i\|^4 = \|r\|^4\,. \tag{6}$$

In addition, for every $j \in S_i^{\mathsf{close}}(\gamma)$, we have

$$\begin{aligned} |q_j - \lambda_j \langle u_i, r\rangle^2| &\leq |\lambda_j| \cdot |\langle u_j, r\rangle^2 - \langle u_i, r\rangle^2| \\ &\leq 2|\lambda_j| \min_{\sigma\in\{\pm 1\}} \|u_i - \sigma\cdot u_j\| \\ &\lesssim |\lambda_j|\, k\sqrt{d}\cdot \min_{\sigma\in\{\pm 1\}} \|v_i - \sigma\cdot v_j\| \\ &\leq T(\gamma)^{1/2}|\lambda_j|\, k\sqrt{d}\,, \end{aligned} \tag{7}$$

where the first and third steps are by Lemma A.6, and the second step is by the fact that we are conditioning on the event of Lemma 2.2. Combining (6) and (7), we conclude that

$$\tau = \left|\sum_{j\in S_i^{\mathsf{close}}(\gamma)} q_j\right| \geq \Lambda\|r\|^4 - T(\gamma)^{1/2}\mathcal{R}\, k\sqrt{d}\,,$$

from which we get the following more quantitative version of the claimed bound.

$$r^\top M_\ell(\lambda_{[k_{\mathsf{res}}]}, \mathbf{u}_{[k_{\mathsf{res}}]})\, r \geq \frac{\Lambda}{2k}\Big(\frac{c^2\gamma^2}{4k^3 d}\Big)^{k}\cdot \|r\|_2^4 - C\mathcal{R}k^2 T(\gamma) - \frac{\sqrt{d}}{2}\Big(\frac{c^2\gamma^2}{4k^3 d}\Big)^{k} T(\gamma)^{1/2}\mathcal{R}\,. \qquad \square$$

To see why such a lower bound on the quadratic form is useful, we show next that it can be used to obtain a net over vectors containing one which is close to some u_i:

Lemma 4.8. *Let $T \subseteq [k]$, and let $\widehat{M}_\ell$ be approximations to $M_\ell(\lambda_T, \mathbf{u}_T)$ satisfying*

$$\|M_\ell(\lambda_T, \mathbf{u}_T) - \widehat{M}_\ell\| \leq \eta$$

for all $\ell \in \{2, \ldots, 2k, 2k+2\}$. Let V be the span of all the top-k singular values of $\widehat{M}_2, \widehat{M}_4, \ldots, \widehat{M}_{2k+2}$. Now let $u_i' = \mathsf{proj}_V(u_i)$ for $i \in T$, and $r_i = u_i - u_i'$.

If for some $i \in T$ and some $\ell \in \{2, \ldots, 2k+2\}$,

$$r_i^\top M_\ell(\lambda_T, \mathbf{u}_T) r_i \geq C_1 \langle r_i, u_i\rangle^2 - C_2,$$

then if we define $\mathcal{S}$ to be an υ-net over unit vectors in V for $\upsilon \asymp (\eta/C_1)^{1/2} + (C_2/C_1)^{1/4}$, then $\mathcal{S}$ contains a vector which is 2υ-close to u_i.

Proof. We will denote $M_\ell(\lambda_T, \mathbf{u}_T)$ by M_ℓ for the rest of the lemma. First let us upper bound $r_i^\top M_\ell r_i$. We have

$$r_i^\top M_\ell r_i = r_i^\top \widehat{M}_\ell r_i + r_i^\top (M_\ell - \widehat{M}_\ell) r_i \leq \lambda_{k+1}(\widehat{M})\|r_i\|^2 + \|M_\ell - \widehat{M}_\ell\|_{\mathsf{op}}\|r_i\|^2 \leq 2\eta\|r_i\|^2$$

Here the first inequality follows from observing that r_i is orthogonal to V which contains the top-k singular subspace of $\widehat{M}_\ell$. The last inequality follows from the following two facts: (i) Weyl's inequality, $\|\lambda(M_\ell) - \lambda(\widehat{M}_\ell)\|_1 \le \eta$ (where $\lambda(A)$ are the eigenvalues of A sorted in decreasing order), and (ii) $\lambda_{k+1}(M_\ell) = 0$ since $\mathsf{rank}(M_\ell) = k$.

Since $\langle r_i, u_i\rangle = \langle r_i, r_i + u_i'\rangle = \|r_i\|^2$, we have

$$2\eta\|r_i\|^2 \ge r_i^\top M_\ell r_i \ge C_1\|r_i\|^4 - C_2.$$

This gives us that $\|r_i\| \le \sqrt{\frac{\eta+\sqrt{\eta^2+C_1C_2}}{C_1}} \le \sqrt{\frac{2\eta}{C_1}} + \sqrt[4]{\frac{C_2}{C_1}}$. Since u_i' is the projection of u to V, this implies that V contains a point close to u_i in ℓ_2 distance.

Let $\mathcal{S}$ be an υ-net over unit vectors in V for $\upsilon \asymp (\eta/C_1)^{1/2} + (C_2/C_1)^{1/4}$. Then by triangle inequality, we know that there exists a point $z \in N$ such that $\|z - u_i\| \le \|z - u_i'\| + \|r_i\| \le 2\upsilon$. □

We now apply Lemma 4.8 using the bound in Lemma 4.7.

Lemma 4.9. *Under the hypotheses of Lemma 4.7, there is an algorithm that takes the parameters $(\widehat{\lambda}, \widehat{\mathbf{u}})$ as well as $\mathrm{poly}(d)\,(\mathcal{R}^2 + \|\widehat{\lambda}\|_1^2)/\xi'^2$ samples from the original function $f_{w,\lambda,\mathbf{u}}$ and outputs a list $\mathcal{S}$ of unit vectors, containing vectors which are υ-close to u_i for every $i \in [k_{\mathsf{res}}]$ for which there exists a gapped scale, where*

$$\upsilon \asymp (\xi'/C_1)^{1/2} + (C_2/C_1)^{1/4} \lesssim \Lambda\,. \tag{8}$$

Proof. Take any such $i \in [k_{\mathsf{res}}]$ with corresponding gapped scale γ. In Lemma 4.8, we will take $T \triangleq [k_{\mathsf{res}}]$, $\widehat{M}_\ell$ given by the empirical estimators from Lemma 4.3, $\eta \triangleq \xi'$, and C_1, C_2 as in Lemma 4.7. Note that by our choice of $T(\gamma)$ in (4), C_2/C_1 is bounded by an arbitrarily small constant multiple of Λ for all $\gamma > 0$, and by our choice of Γ, $(\xi'/C_1)^{1/2} \lesssim \Lambda$. Then we conclude that for υ as in (8), we can recover all i for which there exists a gapped scale to error of order υ. The sample complexity follows from the guarantee of Lemma 4.3. □

4.5 Case 2b: ignoring undetectable neurons

Here we show that if there are undetectable neurons, then we can approximate them by zero:

Lemma 4.10. *Let $\Gamma_{\mathsf{undet}} \subset [k_{\mathsf{res}}] \times [\underline{\gamma}, 1)$ be a set of pairs (i,γ) such that γ is a gapped scale for i and furthermore i is undetectable at scale γ. Additionally, suppose that the sets $\{S_i^{\mathsf{close}}(\gamma)\}_{(i,\gamma)\in\Gamma_{\mathsf{undet}}}$ are all disjoint.*

Then for $S_{\mathsf{rem}} \triangleq [k_{\mathsf{res}}] \setminus \cup_{(i,\gamma)\in\Gamma_{\mathsf{undet}}} S_i^{\mathsf{close}}(\gamma)$,

$$\|f_{\mathsf{res}} - f_{w,\lambda_{S_{\mathsf{rem}}},\mathbf{u}_{S_{\mathsf{rem}}}}\| \lesssim T(\gamma)k^3\sqrt{d} + k\Lambda$$

and thus that

$$\|f_{w,\lambda,\mathbf{u}} - f_{0,\widehat{\lambda},\widehat{\mathbf{u}}} - f_{w,\lambda_{S_{\mathsf{rem}}},\mathbf{u}_{S_{\mathsf{rem}}}}\| \lesssim T(\gamma)k^3\sqrt{d} + k\Lambda + \xi \lesssim \Lambda\,.$$

Proof. Let $(i,\gamma) \in \Gamma_{\mathsf{undet}}$. By the fact that we are conditioning on the event of Lemma 2.2,

$$\|u_i - u_j\| \lesssim T(\gamma)k^2\sqrt{d}$$

We can thus apply Lemma A.2 to the networks $\sum_{j\in S_i^{\mathsf{close}}(\gamma)} \lambda_j|\langle u_j,\cdot\rangle|$ and $(\sum_{j\in S_i^{\mathsf{close}}(\gamma)} \lambda_j)|\langle u_i,\cdot\rangle|$ to get

$$\Big\| \sum_{j\in S_i^{\mathsf{close}}(\gamma)} \lambda_j|\langle u_j,\cdot\rangle| - \Big(\sum_{j\in S_i^{\mathsf{close}}(\gamma)} \lambda_j\Big)\cdot|\langle u_i,\cdot\rangle|\Big\| \lesssim k^2\sqrt{d}\mathcal{R}T(\gamma)\cdot|S_i^{\mathsf{close}}(\gamma)|\,.$$

Additionally, because $|\sum_{j\in S_i^{\mathsf{close}}(\gamma)}\lambda_j|\le\Lambda$,

$$\Big\|\Big(\sum_{j\in S_i^{\mathsf{close}}(\gamma)}\lambda_j\Big)\cdot|\langle u_i,\cdot\rangle|\Big\|\lesssim\Lambda\,.$$

The first part of the lemma follows upon noting that $\sum_{(i,\gamma)\in\Gamma_{\mathsf{undet}}}|S_i^{\mathsf{close}}(\gamma)|\le k$ and that, by definition of f_{res} and disjointness of $S_i^{\mathsf{close}}(\gamma)$ for $(i,\gamma)\in\Gamma_{\mathsf{undet}}$,

$$f_{\mathsf{res}}-f_{w,\lambda_{S_{\mathsf{rem}}},\mathbf{u}_{S_{\mathsf{rem}}}}=\sum_{(i,\gamma)\in\Gamma_{\mathsf{undet}}}\sum_{j\in S_i^{\mathsf{close}}(\gamma)}\lambda_j|\langle u_j,\cdot\rangle|\,.$$

The second part then follows by (2). □

4.6 Combining the cases

We now put together the analysis from the preceding sections. We would like to show that starting from an approximating network $f_{\widehat{\lambda},\widehat{\mathbf{u}}}$ which satisfies (2), after one step of either Case 1 or 2, we end up with an ε-close estimate of the original function $f_{\lambda,\mathbf{u}}$, or otherwise we make progress by approximately learning some new neurons, corresponding to the set of pairs (i,γ) given by $\mathcal{J}$, from the residual network.

First, following Lemma 4.10, let $\Gamma_{\mathsf{undet}}\subseteq\mathcal{J}$ denote the set of pairs (i,γ) for which i is undetectable at scale γ, and recall the definition of $S_{\mathsf{rem}}=[k_{\mathsf{res}}]\setminus\cup_{(i,\gamma)\in\Gamma_{\mathsf{undet}}}S_i^{\mathsf{close}}(\gamma)$. By Lemma 4.10, we can effectively ignore the neurons in $\cup_{(i,\gamma)\in\Gamma_{\mathsf{undet}}}S_i^{\mathsf{close}}(\gamma)$. Among the neurons in S_{rem}, we can use the analysis for Case 2a to recover those in $\mathcal{J}\setminus\Gamma_{\mathsf{undet}}$, i.e. those which have gapped scales at which they are detectable. For these, we can then brute-force over possible weights, resulting in an approximation to the subnetwork given by the neurons in $\mathcal{J}\setminus\Gamma_{\mathsf{undet}}$.

Lemma 4.11. *Let $\mathcal{J}\subseteq S^c$ denote any subset of pairs (i,γ) for which $S_i^{\mathsf{close}}(\gamma)$ are all disjoint and γ is a gapped scale for i. Let $\Gamma_{\mathsf{undet}}\subseteq\mathcal{J}$ denote the set of (i,γ) in $\mathcal{J}$ such that i is undetectable at scale γ.*

Suppose for every $(i,\gamma)\in\mathcal{J}\setminus\Gamma_{\mathsf{undet}}$, we have a vector $\widetilde{u}_i$ satisfying $\|\widetilde{u}_i-u_i\|\le\upsilon$. Then if we define $\mathcal{S}_\lambda$ to be an υ-scale discretization of $[-\mathcal{R},\mathcal{R}]$ and take $S^\star\triangleq[k_{\mathsf{res}}]\setminus\cup_{(i,\gamma)\in\mathcal{J}}S_i^{\mathsf{close}}(\gamma)$, then there exist $\{\widetilde{\lambda}_i\}_{(i,\gamma)\in\mathcal{J}\setminus\Gamma_{\mathsf{undet}}}$ taking values in $\mathcal{S}_\lambda$ such that

$$\left\|f_{w,\lambda,\mathbf{u}}-\left(f_{0,\widehat{\lambda},\widehat{\mathbf{u}}}+\sum_{(i,\gamma)\in\mathcal{J}\setminus\Gamma_{\mathsf{undet}}}\widetilde{\lambda}_i\,|\langle\,\widetilde{u}_i,\cdot\rangle|\right)-f_{w,\lambda_{S^\star},\mathbf{u}_{S^\star}}\right\|\lesssim\Lambda+k^2\mathcal{R}\upsilon.$$

for υ defined in (8).

Proof. Take any $(i,\gamma)\in\mathcal{J}\setminus\Gamma_{\mathsf{undet}}$. Let $\widetilde{\lambda}_i\in\mathcal{S}_\lambda$ be the closest point to $\sum_{j\in S_i^{\mathsf{close}}(\gamma)}\lambda_j$ in $\mathcal{S}_\lambda$. Then by Lemma A.2,

$$\begin{aligned}&\Big\|\sum_{(i,\gamma)\in\mathcal{J}\setminus\Gamma_{\mathsf{undet}}}\sum_{j\in S_i^{\mathsf{close}}(\gamma)}\lambda_j\,|\langle u_j,\cdot\rangle|-\sum_{(i,\gamma)\in\mathcal{J}\setminus\Gamma_{\mathsf{undet}}}\widetilde{\lambda}_i\,|\langle\widetilde{u}_i,\cdot\rangle|\Big\|\\&\qquad\le\sum_{(i,\gamma)\in\mathcal{J}\setminus\Gamma_{\mathsf{undet}}}\Big\|\widetilde{\lambda}_i\,|\langle\widetilde{u}_i,\cdot\rangle|-\sum_{j\in S_i^{\mathsf{close}}(\gamma)}\lambda_j\,|\langle u_j,\cdot\rangle|\Big\|\lesssim k^2\mathcal{R}\upsilon\,.\end{aligned}$$

The lemma then follows by Lemma 4.10, (2), and triangle inequality. □

Lemma 4.12. *At the end of one step of our recursive procedure, either we terminate with a function $\widehat{f} : \mathbb{R}^d \to \mathbb{R}$ such that*

$$\|f_{w,\lambda,\mathbf{u}} - \widehat{f}\| \le \varepsilon$$

or we obtain a subset $S' \supsetneq S$ and $(\overline{\lambda}, \overline{\mathbf{u}}) = (\overline{\lambda}_i, \overline{u}_i)_{i \in S'}$ for which

$$\|f_{w,\lambda,\mathbf{u}} - f_{0,\overline{\lambda},\overline{\mathbf{u}}} - f_{w,\lambda_{(S')^c},\mathbf{u}_{(S')^c}}\| \lesssim k^2 \mathcal{R} \Lambda \,.$$

Proof. At the start of the recursion step, suppose we are in Case 1, then by Lemma 4.5, using $\mathrm{poly}(d)(\mathcal{R}^2 + \|\widehat{\lambda}\|_1^2)/\xi'^2$ samples, we can find a function h that is a linear combination of two ReLUs such that

$$\|f_{w,\lambda,\mathbf{u}} - f_{0,\widehat{\lambda},\widehat{\mathbf{u}}} - h\| \lesssim \mathrm{poly}(d, \mathcal{R}) \cdot (\varepsilon' + \xi') \lesssim \varepsilon \,.$$

In this case, in one of the computation paths of our nondeterministic algorithm, it terminates with an ε-accurate estimator.

Suppose we are not in Case 1, so that by Lemma 4.6 we are in Case 2 and can find some nonempty set $\mathcal{J} \subseteq S^c$ of (i, γ) for which $S_i^{\mathsf{close}}(\gamma)$ are all disjoint and γ is a gapped scale for i. As in Lemma 4.11, let $\Gamma_{\mathsf{undet}} \subseteq \mathcal{J}$ denote the set of $(i, \gamma) \in \mathcal{J}$ such that i is undetectable at scale γ. Then combining Lemma 4.9 and 4.11, using $\mathrm{poly}(d)(\mathcal{R}^2 + \|\widehat{\lambda}\|_1^2)/\xi'^2$ samples for some υ defined in (8) we can (non-deterministically) find $\{\widetilde{\lambda}_i\}_{(i,\gamma) \in \mathcal{J} \setminus \Gamma_{\mathsf{undet}}}$ such that if we add $(\widetilde{\lambda}_i, u_i)_{(i,\gamma) \in \mathcal{J} \setminus \Gamma_{\mathsf{undet}}}$ and $(0, u_i)_{(i,\gamma) \in \Gamma_{\mathsf{undet}}}$ to $(\widehat{\lambda}, \widehat{\mathbf{u}})$ to produce $(\overline{\lambda}, \overline{\mathbf{u}})$, and define S' given by $S' \triangleq S \cup \bigcup_{(i,\gamma) \in \mathcal{J}} \{i\}$, then

$$\|f_{w,\lambda,\mathbf{u}} - f_{0,\overline{\lambda},\overline{\mathbf{u}}} - f_{w,\lambda_{(S')^c},\mathbf{u}_{(S')^c}}\| \le \Lambda + k^2 \mathcal{R} \upsilon \lesssim k^2 \mathcal{R} \Lambda \,. \qquad \square$$

In Section 5, we show that there is a computation path in our nondeterministic algorithm such that the second outcome in Lemma 4.12 happens for at most $O(\log k)$ recursive steps before either $S = [k]$ or we arrive at the first outcome. Formally:

Lemma 4.13. *Given any $(\lambda, \mathbf{u}) \in (\mathbb{R} \times \mathbb{S}^{d-1})^k$, there exists a sequence of sets $\mathcal{J}_1, \dots, \mathcal{J}_q \in [k] \times [\underline{\gamma}, 1)$ such that the following holds. For every $s \in [q]$, let I_s denote the set of $i \in [k]$ for which there exists γ such that $(i, \gamma) \in \mathcal{J}_s$. Then*

1. *$I_1, \dots, I_q$ are disjoint.*
2. *For all $i, j \in [k] \setminus (I_1 \cup \cdots \cup I_q)$, $|v_i - v_j| \le \varepsilon'$ (if $[k] = I_1 \cup \cdots \cup I_q$, this holds vacuously).*
3. *For each $s \in [q]$, all of the subsets $S_i^{\mathsf{close}}(\gamma)$ for $(i, \gamma) \in \mathcal{J}_s$ are disjoint.*
4. *For each $s \in [q]$ and each $(i, \gamma) \in \mathcal{J}_s$, γ is a gapped scale for i.*

We conclude with the main guarantee of this section: some computation path of Algorithm 1 produces an ε-accurate estimate for the original function $f_{\lambda,\mathbf{u}}$.

Lemma 4.14. *Given $\mathrm{poly}(d, 1/\varepsilon, \mathcal{R})^{k^{O(\log^2 k)}}$ samples and runtime, with high probability over the samples and the randomness of the choice of g, there is some computation path in Algorithm 1 in which the algorithm terminates having found a function $\widehat{f} : \mathbb{R}^d \to \mathbb{R}$ such that $\|f_{\lambda,\mathbf{u}} - \widehat{f}\| \le \varepsilon$.*

Proof. Under the second outcome in Lemma 4.12, the L_2 error increases from ξ in (2) to $O(k^2 \mathcal{R} \Lambda)$. Lemma 4.13 ensures that there is some computation path which terminates after this happens $O(\log k)$ times. Recall that we chose $\Lambda \triangleq \mathrm{poly}(d\mathcal{R}/\varepsilon) \cdot \xi^{1/\Theta\left(k^{\log k}\right)}$, so if this increase is repeated

$O(\log k)$ times starting from an initial error of ξ, we end up with an estimator that has error at most $\mathrm{poly}(d\mathcal{R}/\varepsilon) \cdot \xi^{1/\Theta(k^{\log^2 k})}$. In particular, by taking the initial ξ to be

$$\xi = (\varepsilon/\mathcal{R}d)^{\Theta(k^{\log^2 k})}, \tag{9}$$

the final error is bounded by ε as desired. Recall that for a given recursive step, if the initial error in (2) is ξ, then the sample complexity is $\mathrm{poly}(d)(\mathcal{R}^2 + \|\widehat{\lambda}\|_1^2)/\xi'^2 = \mathrm{poly}(d, 1/\varepsilon, \mathcal{R})^{O(k^{\log^2 k})}$ as desired. The number of computation paths is also of this order, as the size of the net $\mathcal{S}_\lambda \times \mathcal{S}$ used in a single recursive step is at most $O(1/\upsilon)^{k^2+k} = \Lambda^{k^2+k} = \mathrm{poly}(d, 1/\varepsilon, \mathcal{R})^{k^{O(\log^2 k)}}$. □

4.7 Power sum estimate

In this section we prove a technical claim which is essential to correctness of the PCA-based procedure described in Case 2a from Section 4.4.

Lemma 4.15. *Let $1 \le k' \le k$, and let $q \in \mathbb{R}^k$ be a vector such that $|\sum_{i=1}^{k'} q_i| \ge \tau$ and $\|q\|_\infty \le R$. If $v \in [-1,1]^k$ satisfies*

1. *$|v_1| \ge \alpha$*
2. *$|v_1 - v_i| \le \beta$ for all $1 \le i \le k'$,*
3. *$|v_i - v_j| \ge \gamma$ for all $1 \le i \le k' < j \le k$.*

for some parameters $0 < \alpha, \beta, \gamma < 1$, then there exists an even integer $0 \le \ell \le 2k$ for which

$$|\langle v^{\odot \ell}, q\rangle| \gtrsim \frac{\tau}{2k}\Big(\frac{\alpha^2\gamma^2}{4k}\Big)^k - CRk(k'-1)\beta$$

for some absolute constant $C > 0$.

To interpret this lemma, it is helpful to first consider the special case where $k' = 1$. In this case, there is a single entry, v_1, of non-negligible magnitude which is separated from all other entries by a margin of γ. The claim is that for any vector q that with a non-negligible first entry, there is some entrywise power of v which has non-negligible correlation with q. Importantly, this holds even if the other entries of v are not well-separated. As such one can interpret this result as some kind of *robust local inverse* for the Vandermonde matrix: even if the $k \times k$ Vandermonde matrix generated by the nodes $v_1^2, \dots, v_k^2$ is ill-conditioned, it is well-conditioned in directions that place sufficient mass on coordinates corresponding to nodes which are separated from the other entries. We also remark that the γ^k scaling is qualitatively tight by the following example: consider $v^{\odot 2} = (1, 1-\gamma, \dots, 1-(k-1)\gamma)$ and $q = (\binom{k-1}{0}, -\binom{k-1}{1}, \binom{k-1}{2}, \dots, (-1)^{k-1}\binom{k-1}{k-1})$. Then one can check that $\langle v^{\odot \ell}, q\rangle = 0$ for all $\ell = 0, \dots, 2k-2$, and for $\ell = k$, this equals $k!\gamma^k$.

For general k', note that the bound in the lemma is non-vacuous provided β scales with γ^k, and this "gap" between γ and γ^k is the central motivation behind our definition of $T(\gamma)$ and the different scales considered in the analysis of Case 2a in Section 4.4.

Proof of Lemma 4.15. We first reduce to the case where $k' = 1$. Consider modifying v, q follows. Remove from v entries $2, \dots, k'$. Also set the first entry of q to be $\sum_{i=1}^{k'} q_i$ and remove from q entries $2, \dots, k'$. As $\|v\|_\infty \le 1$ and $\|q\|_\infty \le R$, this changes every $\langle v^{\odot \ell}, q\rangle$ by at most

$$\sum_{i=1}^{k'} q_i(v_1^\ell - v_i^\ell) = \sum_{i=2}^{k'} q_i(v_1^\ell - v_i^\ell) \lesssim R\ell(k'-1)\beta \lesssim Rk(k'-1)\beta\,.$$

It therefore suffices to prove the lemma for $k' = 1$, so henceforth we specialize to this case.

Suppose to the contrary that for all $\ell = 0, 2, 4, \ldots, 2k-2$, $|\langle v^{\odot \ell}, q\rangle| \le \zeta$ for

$$\zeta \triangleq \frac{\tau}{2k}\Big(\frac{\alpha^2\gamma^2}{4k}\Big)^k.$$

Next, note that

$$\langle v^{\odot \ell}, q\rangle = \sum_{i=1}^k q_i v_i^\ell = v_1^\ell \sum_{i=1}^k q_i (v_i^2/v_1^2)^{\ell/2},$$

so if we define $v_i' \triangleq v_i^2/v_1^2$, then we have that $|\langle v'^{\odot \ell/2}, q\rangle| \le \zeta/v_1^\ell \le \zeta\alpha^{-2k}$. Furthermore, for all $j > 1$, we have $|v_j' - 1| \ge \gamma^2/v_1^2 \ge \gamma^2$. Additionally, $|v_i'| \le 1/\alpha^2$ for all $i \in [k]$.

Define $\varepsilon_i = v_i'^2 - 1$ so that $\varepsilon_1 = 0$ and

$$|\varepsilon_i| \ge \gamma^2$$

for $i > 1$. Then for all $\ell = 0, 2, 4, \ldots, 2k-2$,

$$\langle v'^{\odot \ell/2} - \mathbf{1}, q\rangle = \sum_{i=2}^k q_i((1+\varepsilon_i)^{\ell/2} - 1) = \sum_{i=2}^k q_i \sum_{s=1}^{\ell/2} \binom{\ell/2}{s} \varepsilon_i^s = \sum_{s=1}^{\ell/2} \binom{\ell/2}{s} \sum_{i=2}^k q_i \varepsilon_i^s.$$

As $|\langle v'^{\odot \ell/2} - \mathbf{1}, q\rangle| \le 2\zeta\alpha^{-2k}$ and

$$\Big|\sum_{i=1}^k q_i \varepsilon_i^{\ell/2}\Big| \le |\langle v'^{\odot \ell/2} - \mathbf{1}, q\rangle| + \sum_{s=1}^{\ell/2-1} \binom{\ell/2}{s} \Big|\sum_{i=2}^k q_i \varepsilon_i^s\Big| \le 2\zeta\alpha^{-2k} + 2^k \sum_{s=1}^{\ell/2-1} \Big|\sum_{i=2}^k q_i \varepsilon_i^s\Big|$$

by induction we find that for all $1 \le s < k$,

$$\Big|\sum_{i=2}^k q_i \varepsilon_i^s\Big| \lesssim (2k/\alpha^2)^k \cdot \zeta$$

We also have that

$$\Big|\sum_{i=2}^k q_i\Big| \ge |q_1| - \zeta\alpha^{-2k} \ge \tau - \zeta\alpha^{-2k} > \tau/2\,. \tag{10}$$

We now use Lemma 4.16 to draw a contradiction. Taking $\ell = k-1$ and $z = (\varepsilon_1, \ldots, \varepsilon_k)$ in the lemma, we find that

$$\sum_{i=2}^k q_i \varepsilon_i^{k-1} = \sum_{s=0}^{k-2} (-1)^{k-s} p_{k-1-i}(\varepsilon_2, \ldots, \varepsilon_k) \cdot \sum_{i=2}^k q_i \varepsilon_i^s$$

where here p_s denotes the elementary symmetric polynomial of degree s on $k-1$ variables. We can rearrange to get

$$-p_{k-1}(\varepsilon_2, \ldots, \varepsilon_k) \sum_{i=2}^k q_i = -\sum_{i=2}^k q_i \varepsilon_i^{k-1} + \sum_{s=1}^{k-2} (-1)^{k-s} p_{k-1-i}(\varepsilon_2, \ldots, \varepsilon_k) \cdot \sum_{i=1}^{k-2} q_i \varepsilon_i^s$$

As $|\varepsilon_i| \in [\gamma^2, 1]$, $p_{k-1-s}(\varepsilon_2, \ldots, \varepsilon_k) \le 2^k$ for all s, and $|p_{k-1}(\varepsilon_2, \ldots, \varepsilon_k)| \ge \gamma^{2k}$. So by triangle inequality,

$$\Big|\sum_{i=2}^k q_i\Big| \le k\left(\frac{4k}{\alpha^2\gamma^2}\right)^k \cdot \zeta \le \tau/2,$$

which contradicts (10). □

The above proof used the following basic fact:

Lemma 4.16. *Given $z = (z_1, \dots, z_K)$, let v_ℓ denote the vector $(z_1^\ell, \dots, z_K^\ell)$. Let p_ℓ denote the elementary symmetric polynomial of degree ℓ on K variables. Then*

$$v_K = \sum_{s=0}^{K-1} (-1)^{K-s+1} p_{K-s}(z) \cdot v_s$$

4.8 Validation

Now we have a set of candidate estimators, one of which is guaranteed to be close to the ground truth network in square loss. We will use a validation set of fresh samples to estimate the loss of each of these estimators and pick the best predictor. In order to guarantee that this predictor will have low test loss, we will prove a concentration property of ReLU networks (as in [CKM22]). Notice that for any estimator function $f_{\lambda,\mathbf{u}}$, its difference with the ground truth network f^*, $f_{\lambda,\mathbf{u}} - f^*$ has width $2k$ and is a $2k\mathcal{R}$-Lipschitz function.

Lemma 4.17. *For an arbitrarily given $\delta > 0$, and $t < 4\mathcal{R}^2 k$. Let $F : \mathbb{R}^d \to \mathbb{R}$ be a $2\mathcal{R}$-Lipschitz one-hidden-layer ReLU network with width $2k$. Then, for N i.i.d samples $x_1, x_2, \dots, x_N \sim \mathcal{N}(0, I_d)$, where $N = \Theta\left((\mu + 4\mathcal{R}^2 k)^2 \log(1/\delta)/t^2\right)$. Here $\mu := \mathbb{E}_{x\sim\mathcal{N}(0,I_d)}[F(x)]$. Denote the empirical estimate of the squared loss $\widehat{\sigma}^2 := \frac{1}{N}\sum_{i=1}^N F(x_i)^2$, then with probability $1-\delta$,*

$$\left|\mathbb{E}_{x\sim\mathcal{N}(0,I_d)}\left[F(x)^2\right] - \widehat{\sigma}^2\right| \leqslant t.$$

We defer the proof of this to Appendix B.4.

Since at each step we branch out by a factor of $\mathrm{poly}(d, \mathcal{R}, 1/\varepsilon)^{k^{O(\log^2 k)}}$, and we run for k steps, we have $\mathrm{poly}(d, \mathcal{R}, 1/\varepsilon)^{k^{O(\log^2 k)}}$ predictors to test from. Using the above lemma with $t = \varepsilon$, followed by a union bound, with a validation set of size $\approx k^{O(\log^2 k)}\mathrm{poly}(\mathcal{R}, 1/\varepsilon)\log(d/\delta)$ with probability $1-\delta$, we will find a good predictor up to an additive error of ε. This completes the proof of Theorem 1.1.

5 Terminating in $O(\log k)$ steps

In this section we prove Lemma 4.13, which ensures that there is a successful computation path in our algorithm of length $O(\log k)$. We restate the lemma here for convenience,

Lemma 4.13. *Given any $(\lambda, \mathbf{u}) \in (\mathbb{R}\times\mathbb{S}^{d-1})^k$, there exists a sequence of sets $\mathcal{J}_1, \dots, \mathcal{J}_q \in [k]\times[\underline{\gamma}, 1)$ such that the following holds. For every $s \in [q]$, let I_s denote the set of $i \in [k]$ for which there exists γ such that $(i, \gamma) \in \mathcal{J}_s$. Then*

1. *$I_1, \dots, I_q$ are disjoint.*
2. *For all $i, j \in [k]\backslash(I_1 \cup \cdots \cup I_q)$, $|v_i - v_j| \leq \varepsilon'$ (if $[k] = I_1 \cup \cdots \cup I_q$, this holds vacuously).*
3. *For each $s \in [q]$, all of the subsets $S_i^{\mathsf{close}}(\gamma)$ for $(i, \gamma) \in \mathcal{J}_s$ are disjoint.*
4. *For each $s \in [q]$ and each $(i, \gamma) \in \mathcal{J}_s$, γ is a gapped scale for i.*

Before proving this lemma in Section 5.3, we first identify a particular game in Section 5.1 and upper bound the smallest number of steps needed to win this game. We then show in Section 5.3 how this bound implies an upper bound on the shortest successful computation path in our algorithm.

5.1 Clumping game

This section can be read independently of the rest of this work, and the notational choices here are specific to this section.

Consider the following game. Let $k \in \mathbb{N}$ and let $\tau, \phi \geq 0$. We start with a vector $w \in \mathbb{R}^k_{\geq 0}$ for which $w_1 = w_k = 0$. At every step, we say that a *ϕ-legal move in the (noiseless) clumping game* is a move consisting of the following steps:

1. Select a sequence of indices $1 = i_1 < j_1 \leq i_2 < j_2 \leq \cdots \leq i_m < j_m = k$ such that for every $a \in [m]$, we have that $w_{i_a}, w_{j_a} \leq \tau$ and furthermore at least one of the following two conditions holds:

   - $w_\ell \leq \tau$ for all $i_a < \ell < j_a$.
   - $w_\ell > \max(w_{i_a}, w_{j_a}) + \phi$ for all $i_a < \ell < j_a$.

   In this case, we say that the interval $[i_a, j_a]$ is *good* for w. Note that if $j_a = i_a + 1$ and $w_{i_a}, w_{j_a} \leq \tau$, then it is vacuously true that $[i_a, j_a]$ is good for w.

2. Suppose that the union of the intervals $[i_1, j_1], \ldots, [i_m, j_m]$, *regarded as subsets of* $\mathbb{R}$, is equal to the union of intervals $[i^*_1, j^*_1], \ldots, [i^*_n, j^*_n]$ for $n \leq m$ such that $j^*_a < i^*_{a+1}$ for all a. Then for every a, replace the entries $i^*_a, i^*_a + 1, \ldots, j^*_a$ of w with the single entry $\min_{i^*_a \leq \ell \leq j^*_a} w_\ell$.

3. Update k to be the length of the resulting vector

Steps 1 to 3 altogether count as a single move. The game ends when no more ϕ-legal moves are possible, e.g. when $k = 1$. We remark that the distinction between $\{[i_a, j_a]\}$ versus $\{[i^*_a, j^*_a]\}$ in Step 2 will only be relevant in one place in the proof (see the footnote in the proof of Lemma 5.5). Otherwise, the moves we make will be such that $j_a < i_{a+1}$ for all a, so that there is no difference between $\{[i_a, j_a]\}$ and $\{[i^*_a, j^*_a]\}$.

It is not hard to see that for $\phi = \Omega(1)$ and $\tau \gtrsim \Omega(\log k)$, there always exists a ϕ-legal move that decreases k as long as $k > 1$, so this game will always terminate in at most $k - 1$ moves, and furthermore, by design, the final vector will consist of a single zero entry. The proof of this is essentially identical to the proof of Lemma 4.6.

We will show the following stronger guarantee:

Lemma 5.1. *For sufficiently large absolute constants $c, C > 0$, the following holds. For $\tau = c \log k$, starting at an arbitrary $w \in \mathbb{R}^k_{\geq 0}$ for which $w_1 = w_k = 0$, there is a sequence of at most $C \log k$ moves, each of them 1-legal, after which the game will end, with the final vector consisting of a single zero entry.*

We first introduce some terminology:

Definition 5.2. *Given $r \in \mathbb{N}$ and $i_1, j_1, \ldots, i_m, j_m \in [r]$, we say that intervals $I_1 = [i_1, j_1], \ldots, I_m = [i_m, j_m]$ form a* separated partition $\mathfrak{P}$ of $[r]$ of size m *if*

$$1 = i_1 \leq j_1 < i_2 \leq j_2 < \cdots < i_m \leq j_m = k$$

and furthermore $j_{a+1} > i_a + 1$ for all $1 \leq a < m$. For separated partitions, we will refer to the intervals $\{[j_a + 1, i_{a+1} - 1]\}_{1 \leq a < m}$ as the gaps *of $\mathfrak{P}$.*

Example 5.3. *The intervals $[1, 2], [4, 7], [10, 12]$ form a separated partition of $[12]$, but the intervals $[1, 2], [3, 7], [10, 12]$ do not. For the former, the gaps of the partition are given by the intervals $[3, 3]$ and $[8, 9]$.*

The following trivial observation will be essential to the proof of Lemma 5.1 below:

Lemma 5.4. *Any separated partition of a set $[r]$ has size at most $\lceil r/2 \rceil$.*

Finally, we note that the following can be achieved with two moves.

Lemma 5.5. *Suppose there is a sequence of indices $1 = i_1 \le j_1 \le \cdots \le i_m \le j_m = k$ such that for every $a \in [m]$, we have that $w_{i_a}, w_{j_a} \le \tau$ and furthermore there is some $\tau' \le \tau$ such that for every $i_a \le \ell \le j_a$, we either have $w_\ell \le \tau'$ or $w_\ell > \tau' + 1$. We say that the intervals $[i_a, j_a]$ are* moderate *for w.*

Let w^* be the vector obtained by replacing all of the entries in w indexed by $[i_a, j_a]$ with $\min_{i_a \le \ell \le j_a} w_\ell$, for every a. Then w^* can be obtained from w in two moves.

Proof. For every a, let $[r_1^{(a)}, s_1^{(a)}], \ldots, [r_{m_a}^{(a)}, s_{m_a}^{(a)}]$ denote the separated partition of $[i_a, j_a]$ into intervals such that entries of w corresponding to indices within an interval strictly exceed $\tau' + 1$, and such that entries of w corresponding to indices within a gap of this partition are at most τ'. Then define $i_c^{(a)} = r_c^{(a)} - 1$ and $j_c^{(a)} = s_c^{(a)} + 1$ so that $[i_c^{(a)}, j_c^{(a)}]$ is good for w.

We can make one move using all of the intervals $[i_c^{(a)}, j_c^{(a)}]$.[3] Note that the resulting vector in the next step of the game, call it w', can equivalently be defined by taking w and, within every interval $[i_a, j_a]$ of coordinates, removing all entries of w which exceed $\tau' + 1$ as well as some other entries that are not the minimum entry of w within that interval.

Every interval $[i_a, j_a]$ of coordinates from w corresponds in a natural way to an interval $[i'_a, j'_a]$ of coordinates from w'. Note that within any such interval, the entries of w' are at most τ' by design, so $[i'_a, j'_a]$ is good for w'. We can then make one move using all of the intervals $[i'_a, j'_a]$ to obtain the vector w^* defined in the lemma. □

Lemma 5.6. *If at any point in the game there is exactly one zero entry in the vector, then the game is over.*

Proof. By design, the leftmost and rightmost entries of any vector produced over the course of the game must be zero. So if there is a single zero entry, this means the vector is one-dimensional, and thus the game has ended. □

Proof of Lemma 5.1. Starting with $s = 0$, we inductively define the following objects. Let $k^{(0)} \triangleq k$, $w^{(0)} \triangleq w$, and $u^{(0)} \triangleq w$. We will maintain the invariant that $u^{(s)}$ is a subsequence of $w^{(s)}$ such that all entries outside of $u^{(s)}$ are $> \tau - s + 1$.

Let $\mathfrak{P}_s$ be a separated partition of $[k^{(s)}]$, and denote its size by $k^{(s+1)}$. Suppose that $\mathfrak{P}_s$ consists of intervals $I_1^{(s)}, \ldots, I_{k^{(s+1)}}^{(s)}$ such that:

- Every entry of $u^{(s)}$ indexed by an entry from one of these intervals is $\le \tau - s$
- Every entry of $u^{(s)}$ indexed by an entry from a gap of $\mathfrak{P}_s$ is $> \tau - s$.

Let $1 \le a_1^{(s+1)} < \cdots < a_{k^{(s+1)}}^{(s+1)} \le k^{(s)}$ denote the indices in $I_1^{(s)}, \ldots, I_{k^{(s+1)}}^{(s)}$ over which $u^{(s)}$ is minimized within those intervals (breaking ties arbitrarily). Then define $u^{(s+1)} \in \mathbb{R}_{\ge 0}^{k^{(s+1)}}$ to be the entries of $u^{(s)}$ indexed by $a_1^{(s+1)}, \ldots, a_{k^{(s+1)}}^{(s+1)}$.

[3] This is the only part of the proof in this section where the intervals defining the move are not disjoint as subsets of $\mathbb{R}$, so that there is a distinction between $[i, j]$ and $[i^*, j^*]$ in Step 2.

Finally, we describe how to define $w^{(s+1)}$. First note that each interval $I_\ell^{(s)}$ corresponds to some subset $\{b_{\ell,1}, \dots, b_{\ell,m_\ell}\}$ of the entries of $w^{(s)}$. Consider the intervals $[b_{\ell,1}, b_{\ell,m_\ell}]$ for every ℓ, regarded as subsets of the coordinates of $w^{(s)}$.

We claim that these intervals are all moderate for $w^{(s)}$ in the sense of Lemma 5.5. By the inductive hypothesis, all entries of $w^{(s)}$ outside of $u^{(s)}$ are $> \tau - s + 1$. So within any interval $[b_{\ell,1}, b_{\ell,m_\ell}]$ of coordinates of $w^{(s)}$, the indices which are not $b_{\ell,c}$ for some c are $> \tau - s + 1$, whereas the indices which are $b_{\ell,c}$ for some c are, by assumption on $I_\ell^{(s)}$, at most $\tau - s$. Therefore, the intervals $[b_{\ell,1}, b_{\ell,m_\ell}]$ are moderate for $w^{(s)}$ as claimed.

We can thus make two moves in the game to replace the entries in each interval $[b_{\ell,1}, b_{\ell,m_\ell}]$ of coordinates in $w^{(s)}$ by the single entry $u_\ell^{(s+1)}$. Define $w^{(s+1)}$ to be this new vector. Note that the entries of $w^{(s+1)}$ outside of $u^{(s+1)}$ were either entries of $u^{(s)}$ from among the gaps of $\mathfrak{P}_s$, or entries of $w^{(s)}$ outside of $u^{(s)}$. In the former case, by assumption on $\mathfrak{P}_s$, such entries are $> \tau - s$, and in the latter case, by the inductive hypothesis, such entries are $> \tau - s + 1$. We have thus maintained the invariant that $u^{(s+1)}$ is a subsequence of $w^{(s+1)}$ such that all entries of $w^{(s+1)}$ outside of $u^{(s+1)}$ are $> \tau - (s+1) + 1$.

Note that by Lemma 5.4, $k^{(t)} \le k^{(t-1)}/2$. So after making at most $O(\log k)$ moves as defined above, we end up with a vector $w^{(t)}$ which has exactly one entry which is at most $\tau - t$. If $\tau \ge \Omega(\log k)$, then because there will always be a zero entry within any vector obtained over the course of the game, this means that there is exactly one zero entry in $w^{(t)}$. By Lemma 5.6, this means the game has ended. □

5.2 Noisy clumping game

As we will see in Section 5.3, the steps in our learning algorithm will only *approximately* correspond to moves in the clumping game. More precisely, the former exactly correspond to the following "noisy" version of the clumping game.

Definition 5.7. *Given $w \in \mathbb{R}^k_{\ge 0}$, the vector $w' \in \mathbb{R}^k_{\ge 0}$ is a Δ-perturbation of w if for every $\ell \in [k]$, the following holds:*

- $w'_\ell \le w_\ell$
- *If additionally $w_\ell > 1$, then $w'_\ell \ge w_\ell - \Delta$.*

Given $k \in \mathbb{N}$, $\tau, \phi \ge 0$, and starting vector $w \in \mathbb{R}^k_{\ge 0}$ for which $w_1 = w_k = 0$, we say that a *ϕ-legal move in the **noisy** clumping game* is a move consisting of the following four steps. The first three steps are identical to those of the noiseless clumping game. Then in the fourth step,

4. Replace w with an arbitrary (possibly adversarially chosen) $1/100k$-perturbation of w.

Lemma 5.9 ensures that a 1-legal strategy for the noiseless clumping game can be converted into a 0.99-legal strategy for the noisy clumping game:

Lemma 5.8. *Let $w \in \mathbb{R}^k_{\ge 0}$ be any vector for which $w_1 = w_k = 0$. If there is a sequence of N 1-legal moves in the noiseless clumping game starting from w after which the game ends with the final vector consisting of a single zero entry, then there is a sequence of N $(1-\Delta)$-legal moves starting from w in the noisy clumping game after which the game ends with the final vector consisting of a single zero entry.*

This follows immediately by inducting on k and repeatedly applying the following:

Lemma 5.9. *Let $w \in \mathbb{R}^k_{\geq 0}$ be any vector for which $w_1 = w_k = 0$, and let w' be any Δ-perturbation of w. Any 1-legal move in the noiseless clumping game starting from w is also a $(1-\Delta)$-legal move in the noisy clumping game starting from w'.*

Let u, u' be the vectors resulting from the former and latter respectively. Then u' is a $(\Delta + 1/100k)$-perturbation of u.

Proof. Suppose the 1-legal move in the noiseless clumping game is specified by intervals $\{[i_a, j_a]\}_{a\in[m]}$.

Certainly for any $a \in [m]$, if $w_{i_a}, w_{j_a} \leq \tau$, then $w'_{i_a}, w'_{j_a} \leq \tau$. Likewise, if $w_\ell \leq \tau$ for all $i_a < \ell < j_a$, then $w'_\ell \leq \tau$ for all $i_a < \ell < j_a$. Otherwise, if $w_\ell > \max(w_{i_a}, w_{j_a}) + 1$ for all $i_a < \ell < j_a$, then by the definition of Δ-perturbation, $w'_\ell \in [w_\ell - \Delta, w_\ell]$, so $w'_\ell > \max(w_{i_a}, w_{j_a}) + 1 - \Delta \geq \max(w'_{i_a}, w'_{j_a}) + 1$ for all $i_a < \ell < j_a$. We conclude that $\{[i_a, j_a]\}_{a\in[m]}$ is a $(1-\Delta)$-legal move. The last part of the lemma follows from the fact that a $1/100k$-perturbation of a Δ-perturbation is a $(\Delta + 1/100k)$-perturbation. □

5.3 Relating the noisy clumping game to Lemma 4.13

We show that any sequence of $\mathcal{J}_1, \ldots, \mathcal{J}_q$ satisfying the four conditions of Lemma 4.13 corresponds to a sequence of 0.99-legal moves in the noisy clumping game, and vice versa, after which we can conclude the proof of Lemma 4.13 by invoking Lemma 5.1 and Lemma 5.8.

First, we need the following definition:

Definition 5.10. *Let c be the constant in the exponent of γ/d in the definition of $T(\gamma)$ in (4). Given $\gamma \in [0, 1)$, define the* level of γ, *denoted $L(\gamma)$, by*

$$L(\gamma) \triangleq \frac{0.9}{\ln(ck)} \ln\Big(1 + \frac{(ck-1)\ln(\varepsilon'/\gamma)}{ck \ln d + \ln(\mathcal{R}^2/\Lambda^{10}) + (ck-1)\ln(1/\varepsilon')}\Big)$$

where Λ is defined in (3) and ξ therein is given by (9). This is clearly monotonically increasing as γ decreases.

The function L is chosen so that

$$L(T(\gamma)) = 0.9 + L(\gamma)\,.$$

In particular, the level of $T^{(n)}(\varepsilon')$ is precisely $0.9n$, where $T^{(n)}$ denotes n-fold composition of T.

Observation 5.11. *For any $0 \leq \gamma_1 \leq \cdots \leq \gamma_s$, if $L(\gamma_s) > 1$, then*

$$L(\gamma_s) - O(1/(k \ln d)) \leq L(\gamma_1 + \cdots + \gamma_s) \leq L(\gamma_s)\,.$$

The latter bound also holds if $L(\gamma_s) \leq 1$.

Proof. That $L(\gamma_1 + \cdots + \gamma_s) \leq L(\gamma_s)$ follows immediately from the fact that L is a monotonically decreasing function. For the other bound, for convenience, denote $\beta \triangleq ck \ln d + \ln(\mathcal{R}^2/\Lambda^{10} + (ck-1)\ln(1/\varepsilon')$ and note that $L(\gamma_s) \geq 1$ implies that $\ln(1 + (ck-1)\ln(\varepsilon'/\gamma)/\beta) \geq ck$ so that

$$\ln \frac{1 + \ln(1/k\gamma)\cdot(ck-1)/\beta}{1 + \ln(1/\gamma)\cdot(ck-1)/\beta} \geq \ln\Big(1 - \frac{\ln(k)\cdot(ck-1)/\beta}{ck}\Big) \geq 1 - O(\ln k/(k \ln d))\,,$$

so

$$L(\gamma_1 + \cdots + \gamma_s) \geq L(k\gamma_s) \geq L(\gamma_s) - O(1/(k \ln d))\,.$$

□

Proof of Lemma 4.13. Take the "k" in Section 5.1 to be $k+1$, and let the initial vector "w" be defined as follows. As specified in Section 5.1, we take its first and last entries to be 0. For $1 < i < k$, let the i-th entry of "w" from Section 5.1 be given by

$$w_i \triangleq L(v_i - v_{i-1}) .$$

Define

$$\tau \triangleq L(\underline{\gamma}) = \Theta(\log k) .$$

For $\phi = 0.99$, consider any first move $\{[i_a, j_a]\}$ in the noisy clumping game, starting from the vector w.

Suppose first that this move does not consist of $\{[1, k+1]\}$. We show that this move corresponds to a choice of $\mathcal{J}$ satisfying the conditions that

(i) All the subsets $S_i^{\mathsf{close}}(\gamma)$ for $(i,\gamma) \in \mathcal{J}$ are disjoint.

(ii) For each $(i,\gamma) \in \mathcal{J}$, γ is a gapped scale for i.

For each $a \in [m]$,

(A) If $w_\ell \le \tau$ for all $i_a \le \ell \le j_a$: by definition of w, this means that $L(v_\ell - v_{\ell-1}) \le \tau$ for all $\max(i_a, 2) \le \ell \le \min(j_a, k)$. So $v_{\max(i_a-1,1)}, \dots, v_{\min(j_a,k)}$ are all separated by a distance of at least $\underline{\gamma}$. This means that $S_\ell^{\mathsf{close}}(\underline{\gamma}) = \{\ell\}$ while $S_\ell^{\mathsf{far}}(\underline{\gamma}) = [k] \backslash \{\ell\}$, so $\underline{\gamma}$ is a gapped scale for every $i_a \le \ell < j_a$. Add $(\ell, \underline{\gamma})$ for all $i_a \le \ell < j_a$ to $\mathcal{J}$.

(B) If $w_\ell > \max(w_{i_a}, w_{j_a}) + 0.99$ for all $i_a < \ell < j_a$: this means that

$$v_\ell - v_{\ell-1} \ll \min(T(v_{i_a} - v_{i_a-1}), T(v_{j_a} - v_{j_a-1}))$$

for all $\max(i_a, 2) \le \ell \le \min(j_a, k)$. In fact, because of the margin between $\phi = 0.99$ and the constant 0.9 in the definition of L and by Observation 5.11, this ensures that

$$v_\ell - v_{\ell'} \ll \min(T(v_{i_a} - v_{i_a-1}), T(v_{j_a} - v_{j_a-1}))$$

for all $i_a \le \ell < j_a$.

Let γ be such that $L(\gamma) = \min(w_{i_a}, w_{j_a})$. If the minimum is achieved by the former (resp. the latter), then γ is a gapped scale for i_a (resp. j_a) and $S_{i_a}^{\mathsf{close}}(\gamma) = \{i_a, \dots, j_a - 1\}$ (resp. $S_{j_a}^{\mathsf{close}}(\gamma) = \{i_a, \dots, j_a - 1\}$). Add (i_a, γ) (resp. (j_a, γ)) to $\mathcal{J}$.

It is clear from the above construction of $\mathcal{J}$ that the two conditions (i) and (ii) hold.

Finally, we need to relate the removal of neurons indexed by $\cup_{(i,\gamma)\in\mathcal{J}} S_i^{\mathsf{close}}(\gamma)$ from $[k]$ to Steps 2 and 4 of the clumping game. Indeed, for each $(i,\gamma) \in \mathcal{J}$, if the pair came from case (A) above and corresponds to an index ℓ for which $i_a \le \ell < j_a$, then $S_i^{\mathsf{close}}(\gamma) = \{\ell\}$. Recall that for all $[i_a, j_a]$ in case (A), there is a pair in $\mathcal{J}$ corresponding to each such ℓ. On the other hand, if (i,γ) instead came from case (B) above and $[i_a, j_a]$ is the corresponding interval from the move of the noisy clumping game, then $S_i^{\mathsf{close}}(\gamma) \subset [k]$ is the set $\{i_a, \dots, j_a - 1\}$. Putting things together, we conclude that $\cup_{(i,\gamma)\in\mathcal{J}} S_i^{\mathsf{close}}(\gamma) = \cup_a \{i_a, \dots, j_a - 1\}$. So if $[i_1^*, j_1^*], \dots, [i_n^*, j_n^*]$ are the intervals defined in Step 2 of the game, then $\cup_{(i,\gamma)\in\mathcal{J}} S_i^{\mathsf{close}}(\gamma) = \cup_a \{i_a^*, \dots, j_a^* - 1\}$.

Note that for every $1 \le a < n$, $L(v_{j_a^*+1} - v_{i_a^*-1}) \le L(\min_{i_a^* \le \ell \le j_a^*} w_\ell)$ and additionally, if $L(\min_{i_a^* \le \ell \le j_a^*} w_\ell) \ge 1$, then $L(v_{j_a^*+1} - v_{i_a^*-1}) \ge L(\min_{i_a^* \le \ell \le j_a^*} w_\ell) - O(1/(k \log d))$. For d greater than a sufficiently large constant, the quantity $O(1/(k \log d))$ is at most $1/100k$, so if $\ell_1, \dots, \ell_s$ are

the neurons remaining after the first step of the algorithm, the vector with entries consisting of $L(v_{\ell_a} - v_{\ell_{a-1}})$ is a valid $1/100k$ perturbation of the vector obtained from Step 2 of the clumping game.

We have thus shown that any initial 0.99-legal move $\{[i_a, j_a]\}$ in the noisy clumping game that does not consist solely of the interval $[1, k+1]$ corresponds to a valid set $\mathcal{J}$ of neurons that can be learned and removed from consideration in the first iteration of our recursive learner, and moreover the sequence of pairwise separations between successive v_i's associated to the remaining neurons corresponds to the new vector w in Step 4 from making the move $\{[i_a, j_a]\}$ in the noisy clumping game. See Figure 2 for an illustration of this correspondence.

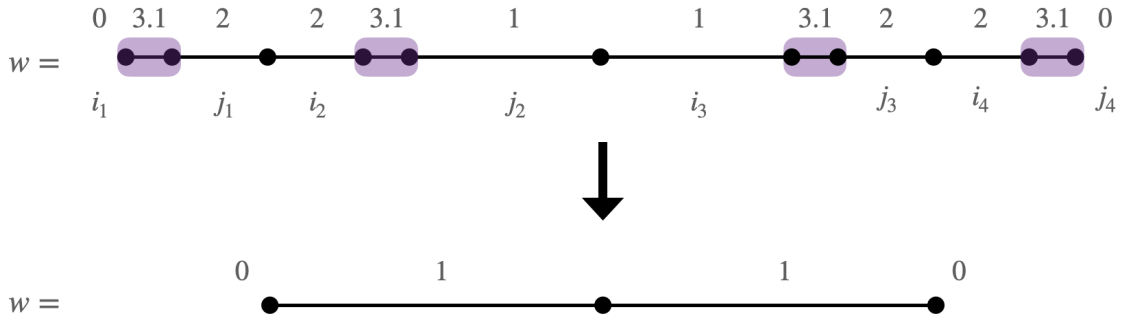

Figure 2: Example of a 1-legal move in the (noiseless) clumping game, with $\tau = 3$. Initially, the vector is given by $w = (0, 3.1, 2, 2, 3.1, 1, 1, 3.1, 2, 2, 3.1, 0)$, and the move is given by $(i_1, j_1) = (1, 3), (i_2, j_2) = (4, 6), (i_3, j_3) = (7, 9), (i_4, j_4) = (10, 12)$. The vector resulting from this move is $(0, 1, 0)$. Below the entries of w is a sequence of points v_i such that the distance between the $(i-1)$-st and i-th point is γ satisfying $L(\gamma) = w_i$. The highlighted regions correspond to clumps of neurons for which there is a gapped scale and which can thus be learned using the analysis in Case 2a in Section 4.4. After these neurons are learned and subtracted out of the network, the remaining three neurons have pairwise separations γ_1, γ_2 for which $(L(\gamma_1), L(\gamma_2)) \approx (1, 1)$.

The above reasoning can then be applied in an identical fashion to subsequent moves in the clumping game / subsequent iterations of the recursive learner, provided the move does not consist solely of the interval $[1, k+1]$.

Lemma 5.1 and Lemma 5.8 ensure the existence of a sequence of 0.99-legal moves in the noisy clumping game at the end of which the game ends with the vector w consisting of a single zero entry. The last move in this sequence must be the move $[1, k+1]$. If $\ell_1, \dots, \ell_s \in [k]$ are the only remaining neurons prior to this final move, the move $[1, k+1]$ is only 0.99-legal provided that $L(v_{\ell_a} - v_{\ell_{a-1}}) > 0.99$, but in this case $|v_{\ell_a} - v_{\ell_b}| \le k \cdot T(1) \ll \varepsilon'$ for all $a, b \in [s]$, as desired. □

6 Conclusion and Future Directions

In this paper, we provided the first PAC learning algorithm that learns narrow (constant k) one-hidden-layer neural networks in polynomial time. Our algorithm used random contractions of higher order moment tensors and subsequently performed an iterative procedure to repeatedly learn some clumps of neurons. This allowed us to avoid depending on the condition number of the weight matrix, which was a core assumption in several prior works.

One obvious drawback of our technique is the unsatisfactory dependence on k in our runtime $d^{k^{\mathcal{O}(\log^2 k)}}$. Note that known CSQ hardness results rule out the possibility of improving to $d^{o(k)}$

for our algorithm, but we leave as an interesting open question closing the gap between our upper bound and these lower bounds. Furthermore, harnessing the power of non-CSQ style approaches as in [CM20, CKM22] could potentially allow us to get an algorithm that runs in time $\mathrm{poly}(d)\cdot(1/\varepsilon)^{h(k)}$ for some function h.

Another worthwhile direction is to investigate other applications of the power sum estimate technique proposed in our work to tensor problems in the absence of separation conditions.

References

[AZLL19] 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.

[BJW19] 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.

[CKM21] Sitan Chen, Adam R Klivans, and Raghu Meka. Efficiently learning any one hidden layer relu network from queries. *arXiv preprint arXiv:2111.04727*, 2021.

[CKM22] 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.

[CLLZ22] Sitan Chen, Jerry Li, Yuanzhi Li, and Anru R Zhang. Learning polynomial transformations. *arXiv preprint arXiv:2204.04209*, 2022.

[CM20] Sitan Chen and Raghu Meka. Learning polynomials of few relevant dimensions. *arXiv preprint arXiv:2004.13748*, 2020.

[Dan17] Amit Daniely. Sgd learns the conjugate kernel class of the network. *CoRR*, abs/1702.08503, 2017.

[DGK+20] Ilias Diakonikolas, Surbhi Goel, Sushrut Karmalkar, Adam R Klivans, and Mahdi Soltanolkotabi. Approximation schemes for relu regression. In *Conference on Learning Theory*, 2020.

[DK20] 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.

[DKKZ20] 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, 2020.

[DV20] Amit Daniely and Gal Vardi. Hardness of learning neural networks with natural weights. *arXiv preprint arXiv:2006.03177*, 2020.

[GGJ+20] 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.

[GK19] 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.

[GKKT17] 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.

[GKLW18] Rong Ge, Rohith Kuditipudi, Zhize Li, and Xiang Wang. Learning two-layer neural networks with symmetric inputs. In *International Conference on Learning Representations*, 2018.

[GKM18] Surbhi Goel, Adam R. Klivans, and Raghu Meka. Learning one convolutional layer with overlapping patches. In *ICML*, volume 80, pages 1778–1786. PMLR, 2018.

[GLM18] 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*, 2018.

[GMOV18] 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.

[JSA15] 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.

[LMZ20] 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.

[LY17] 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.

[MR18] Pasin Manurangsi and Daniel Reichman. The computational complexity of training relu (s). *arXiv preprint arXiv:1810.04207*, 2018.

[MSS16] Tengyu Ma, Jonathan Shi, and David Steurer. Polynomial-time tensor decompositions with sum-of-squares. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 438–446. IEEE, 2016.

[Sha18] Ohad Shamir. Distribution-specific hardness of learning neural networks. *Journal of Machine Learning Research*, 19(32):1–29, 2018.

[SJA16] 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.

[Sol17] Mahdi Soltanolkotabi. Learning relus via gradient descent. In *Advances in neural information processing systems*, pages 2007–2017, 2017.

[SSSS17] Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. Failures of gradient-based deep learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3067–3075, 2017.

[VW19] Santosh Vempala and John Wilmes. Gradient descent for one-hidden-layer neural networks: Polynomial convergence and sq lower bounds. In *COLT*, volume 99, 2019.

[ZLJ16] 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.

[ZPS17] Qiuyi Zhang, Rina Panigrahy, and Sushant Sachdeva. Electron-proton dynamics in deep learning. *CoRR*, abs/1702.00458, 2017.

[ZSJ+17] 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.

[ZYWG19] 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.

A Deferred Preliminaries

A.1 Hermite polynomials

Recall the definition of the probabilist's Hermite polynomials:

$$H_n(x) = (-1)^n e^{\frac{x^2}{2}} \cdot \frac{d^2}{dx^2} e^{-\frac{x^2}{2}}.$$

Under this definition, the first four Hermite polynomials are

$$H_0(x) = 1, \quad H_1(x) = x, \quad H_2(x) = x^2 - 1, \quad H_3(x) = x^3 - 3x.$$

The Hermite polynomials comprise an orthogonal basis of the Hilbert space $\mathcal{L}^2(\mathbb{R}, \omega)$ where ω is the standard Gaussian measure on $\mathbb{R}$. In this function space, the inner product is defined as $\langle f, g\rangle = \mathbb{E}_{x\sim\mathcal{N}(0,1)}[f(x)g(x)]$. Under this inner product, we have

$$\langle H_m, H_n\rangle = \mathbb{E}_{x\sim\mathcal{N}(0,1)} H_m(x)H_n(x) = m! \cdot \mathbb{I}[m = n].$$

The *normalized* probabilist's Hermite polynomials are given by $\widehat{H}_\ell(x) \triangleq \frac{1}{\sqrt{\ell!}} H_\ell(x)$; these comprise an orthonormal basis of $\mathcal{L}^2(\mathbb{R}, \omega)$. Finally, given $x \in \mathbb{R}^d$, we define the *Hermite tensor* $S_\ell(x) \in (\mathbb{R}^d)^{\otimes \ell}$ to be the tensor whose $(i_1, \ldots, i_\ell)$-th entry is given as follows. Suppose that element $j \in [d]$ appears among $i_1, \ldots, i_\ell$ a total of n_j times. Then the $(i_1, \ldots, i_\ell)$-th entry of S_ℓ is given by $\prod_{j=1}^d \widehat{H}_{n_j}(x_j)$.

A.2 Measuring distance between networks

Here, we will show that parameter closeness implies closeness in the square loss, which also implies the closeness between moment tensors.

Lemma A.1 (Lemma 3.3 from [CKM21])**.** *For any unit vectors u, u',*

$$\mathbb{E}\left[\mathsf{relu}(\langle u, x\rangle) - \mathsf{relu}(\langle u', x\rangle)\right]^2 \le \frac{5}{6}\|u - u'\|_2^2.$$

Lemma A.2. *For two 2-layer ReLU networks $f_{\lambda,\mathbf{u}}, f_{\lambda',\mathbf{u}'}$ with the same width k and for which $\|\lambda\|_1, \|\lambda'\|_1 \le \mathcal{R}$ for some $\mathcal{R} > 0$, we have*

$$\|f_{\lambda,\mathbf{u}} - f_{\lambda',\mathbf{u}'}\|_2 \lesssim k \max(1, \mathcal{R}) \cdot d_{\mathsf{param}}((\lambda, \mathbf{u}), (\lambda', \mathbf{u}')).$$

Proof. By AM-GM, for any permutation π:

$$\begin{aligned}
&\|f_{\lambda,\mathbf{u}} - f_{\lambda',\mathbf{u}'}\|_2^2 = \mathbb{E}\left(\sum_{i=1}^k \lambda_i \cdot \mathsf{relu}(\langle u_i, x\rangle) - \lambda'_{\pi(i)} \cdot \mathsf{relu}(\langle u'_{\pi(i)}, x\rangle)\right)^2 \\
&= \mathbb{E}\left(\sum_{i=1}^k (\lambda_i - \lambda'_{\pi(i)}) \cdot \mathsf{relu}(\langle u_i, x\rangle) + \sum_{i=1}^k \lambda'_{\pi(i)} \cdot \Big(\mathsf{relu}(\langle u_i, x\rangle) - \mathsf{relu}(\langle u'_{\pi(i)}, x\rangle)\Big)\right)^2 \\
&\leqslant (2k) \cdot \sum_{i=1}^k \mathbb{E}\left[(\lambda_i - \lambda'_{\pi(i)})^2 \cdot \mathsf{relu}(\langle u_i, x\rangle)^2\right] + (2k) \cdot \sum_{i=1}^k \mathbb{E}\left[\lambda_i'^2 \cdot \Big(\mathsf{relu}(\langle u_i, x\rangle) - \mathsf{relu}(\langle u'_{\pi(i)}, x\rangle)\Big)^2\right] \\
&\leqslant k \sum_{i=1}^k (\lambda_i - \lambda'_{\pi(i)})^2 + \frac{5k}{3} \sum_{i=1}^k \lambda'^2_{\pi(i)} \cdot \|u_i - u'_{\pi(i)}\|_2^2 \\
&\leqslant k^2 \cdot d_{\mathsf{param}}((\lambda, \mathbf{u}), (\lambda', \mathbf{u}'))^2 + \frac{5k^2}{3}\mathcal{R}^2 \cdot d_{\mathsf{param}}((\lambda, \mathbf{u}), (\lambda', \mathbf{u}'))^2 \\
&\leqslant k^2(1 + 2\mathcal{R}^2) \cdot d_{\mathsf{param}}((\lambda, \mathbf{u}), (\lambda', \mathbf{u}'))^2,
\end{aligned}$$

where in the third step we used Lemma A.1. Therefore, we conclude that

$$\|f_{\lambda,\mathbf{u}} - f_{\lambda',\mathbf{u}'}\|_2 \leqslant k\sqrt{1+2R^2} \cdot d_{\mathsf{param}}((\lambda,\mathbf{u}),(\lambda',\mathbf{u}')),$$

as claimed. □

Lemma A.3. *For the 2-norm of $f_{\lambda,\mathbf{u}}$ and the Frobenius norm of its moment tensor, it holds that:*

$$\|T_\ell(\lambda,\mathbf{u})\|_F \leq \sqrt{2\pi} \cdot d^{\ell/2} \cdot \|f_{\lambda,\mathbf{u}}\|_2.$$

Proof. To prove this equation, we firstly need to decompose the ReLU activation function $\mathsf{relu}(\cdot)$ with the Hermite polynomials. From Lemma A.2 of [GGJ+20], we have:

$$\mathsf{relu}(x) = \sum_{k=0}^{\infty} c_k H_k(x),$$

where the coefficients are

$$c_0 = \frac{1}{\sqrt{2\pi}}, c_1 = \frac{1}{2},\ c_{2k} = \frac{(-1)^{k+1}(2k-3)!!}{\sqrt{2\pi}\cdot(2k)!}\ (k \geqslant 1)\ \text{ and }\ c_{2k+1} = 0.$$

Note that, these coefficients are slightly different from [GGJ+20] since our definition of H_n is unnormalized (which means their H_n stands for our $\widehat{H}_n$).

Therefore, we can express $T_\ell(\lambda,\mathbf{u})$ as:

$$\mathbb{E}\left[f_{\lambda,\mathbf{u}}(x)\cdot H_\ell(x)\right] = \ell!\cdot c_\ell \sum_{i=1}^{k} \lambda_i u_i^{\otimes\ell} = \ell!\cdot c_\ell T_\ell(\lambda,\mathbf{u}).$$

Now, we take the Frobenius norm of both sides, we have:

$$\begin{aligned}
\ell!\cdot\frac{(\ell-3)!!}{\sqrt{2\pi}\ell!}\cdot\|T_\ell(\lambda,\mathbf{u})\|_F = \|\mathbb{E}\left[f_{\lambda,\mathbf{u}}(x)\cdot H_\ell(x)\right]\|_F &= \sqrt{\sum_{\alpha\in[d]^l}\left(\mathbb{E}\left[f_{\lambda,\mathbf{u}}(x)\cdot H_{\ell,\alpha}(x)\right]\right)^2} \\
&\leqslant \sqrt{\sum_{\alpha\in[d]^l}\mathbb{E}[f_{\lambda,\mathbf{u}}(x)^2]\cdot\mathbb{E}\left[H_{\ell,\alpha}(x)^2\right]} = \|f_{\lambda,\mathbf{u}}\|_2\cdot\sqrt{\mathbb{E}\|H_\ell(x)\|_F^2} \\
&< \|f_{\lambda,\mathbf{u}}\|_2\cdot\sqrt{d^\ell\cdot\ell!}
\end{aligned}$$

Since $(\ell-3)!! < \sqrt{\ell!}$, we can conclude that:

$$\|T_\ell(\lambda,\mathbf{u})\|_F \leqslant \sqrt{2\pi}\cdot d^{\ell/2}\|f_{\lambda,\mathbf{u}}\|_2.$$

□

Lemma A.4. *For any unit vector $g\in\mathbb{S}^{d-1}$, it holds that:*

$$\|M_\ell^g(\lambda,\mathbf{u}) - M_\ell^g(\lambda',\mathbf{u}')\|_{\mathsf{op}} \leq \|T_\ell(\lambda,\mathbf{u}) - T_\ell(\lambda',\mathbf{u}')\|_F.$$

Proof. Notice that:

$$M_\ell^g(\lambda, \mathbf{u}) = T_\ell(\lambda, \mathbf{u})[g, \dots, g, :, :], \quad M_\ell^g(\lambda', \mathbf{u}') = T_\ell(\lambda', \mathbf{u}')[g, \dots, g, :, :].$$

Then, for any unit vector $v \in \mathbb{S}^{d-1}$, we have:

$$\begin{aligned}
&v^\top \left(M_\ell^g(\lambda, \mathbf{u}) - M_\ell^g(\lambda', \mathbf{u}')\right) v = \left(T_\ell(\lambda, \mathbf{u}) - T_\ell(\lambda', \mathbf{u}')\right)[g, \dots, g, v, v] \\
&= \langle T_\ell(\lambda, \mathbf{u}) - T_\ell(\lambda', \mathbf{u}'), g \otimes \ldots \otimes g \otimes v \otimes v \rangle \\
&\leqslant \left\|T_\ell(\lambda, \mathbf{u}) - T_\ell(\lambda', \mathbf{u}'\right\|_F \cdot \|g \otimes \ldots \otimes g \otimes v \otimes v\|_F \\
&= \left\|T_\ell(\lambda, \mathbf{u}) - T_\ell(\lambda', \mathbf{u}'\right\|_F \cdot \left(\|g\|_2^{l-2} \|v\|_2^2\right) \leqslant \left\|T_\ell(\lambda, \mathbf{u}) - T_\ell(\lambda', \mathbf{u}')\right\|_F,
\end{aligned}$$

which leads to the conclusion that

$$\|M_\ell^g(\lambda, \mathbf{u}) - M_\ell^g(\lambda', \mathbf{u}')\|_{\mathsf{op}} \leq \|T_\ell(\lambda, \mathbf{u}) - T_\ell(\lambda', \mathbf{u}')\|_F.$$

□

A.3 Proofs for anti-concentration

We use the following standard bound:

Lemma A.5. *Given unit vector $u \in \mathbb{S}^{d-1}$, if g is a random unit vector, then with probability at least $1 - \delta$,*

$$\sqrt{d}|\langle u, g\rangle| \in [c\delta, c'\sqrt{\ln(2/\delta)}]$$

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

Proof. We can write g as $h/\|h\|$ for $h \sim \mathcal{N}(0, \mathrm{Id})$. Then $\langle u, h\rangle \sim \mathcal{N}(0, \|v\|)$ and we have

$$\begin{aligned}
&\Pr[|\langle u, h\rangle| \geq \sqrt{2\ln(2/\delta)}] \leq \delta/2 \\
&\Pr[|\langle u, h\rangle| \leq (\delta/2)/\sqrt{2/\pi}] \leq \delta/2\,.
\end{aligned}$$

Additionally, $\Pr[|\|h\| - \sqrt{d}| \geq \sqrt{d}/2] \leq \exp(-\Omega(d))$. The lemma follows by a union bound. □

We can now complete the proofs of the two lemmas from Section 2.2:

Proof of Lemma 2.2. Take any $u \triangleq \frac{1}{\|u_i + \sigma \cdot u_j\|} \cdot (u_i + \sigma \cdot u_j)$ and note that by Lemma A.5 applied to $\delta = 1/10k^2$, we have $|\langle u, g\rangle| \in \left[\frac{c}{10k^2\sqrt{d}}, \frac{c'\sqrt{\ln(20k^2)}}{\sqrt{d}}\right]$. The lemma follows by a union bound over all $i, j \in [k]$ and $\sigma \in \{\pm 1\}$. □

Proof of Lemma 2.3. Take u in Lemma A.5 to be u_i and $\delta = 1/10k$ to conclude that $\sqrt{d}|\langle u, g\rangle| \geq c/10k$. The lemma follows by a union bound over i. □

Finally, we record the following elementary inequality:

Lemma A.6. *For any $a, b \in \mathbb{R}$ satisfying $|a|, |b| \leq 1$,*

$$\min_{\sigma \in \{\pm 1\}} |a - \sigma b|^2 \leq |a^2 - b^2| \leq 2 \min_{\sigma \in \{\pm 1\}} |a - \sigma b|\,.$$

Proof. Both bounds are immediate from the fact that $|a^2 - b^2| = |a - b| \cdot |a + b|$. □

B Deferred Proofs From Section 4

B.1 Proof of Lemma 4.2

This was essentially, e.g., in Corollary 42 of [DK20]. Note that while that work considered the case of positive λ_i, their proof of Corollary 42 does not use positivity. Additionally, their guarantee is stated for all even ℓ only because their algorithm only makes use of even ℓ, even though their proof of Corollary 42 applies equally well to $\ell = 1$.

Additionally, their guarantee is stated in terms of relu activation instead of absolute value activation. But note that $|z| = \mathsf{relu}(z) + \mathsf{relu}(-z)$, so because c_ℓ is the ℓ-th normalized probabilist's Hermite coefficient of $\mathsf{relu}(\cdot)$ and satisfies $c_\ell = \Theta(\ell^{-5/2})$ (see e.g. [GGJ+20, Lemma A.2]). we conclude that the corresponding Hermite coefficient for $|\cdot|$ is $2c_\ell$, so $\mathbb{E}[\widehat{T}] = T_\ell$. The claim for $\ell = 1$ follows similarly.

B.2 Proof of Lemma 4.3

The degree-1 Hermite coefficients of $f_{0,\widehat{\lambda},\mathbf{u}}$ is zero, while the degree-1 Hermite coefficients of $f_{w,\lambda,\mathbf{u}}$ are given by w, so the expectation of $\frac{1}{N}\sum_{a=1}^N (y_a - f_{0,\lambda,\mathbf{u}}(x_a))x_a$ is w. By Lemma 4.2, the deviation between the empirical mean and the population mean is bounded by $\xi'/2$ provided $N \geq \mathrm{poly}(d)\,(\mathcal{R}^2 + \|\widehat{\lambda}\|_1^2)/\xi'^2$. This establishes the first bound.

Note that by Lemmas A.2, A.3, and A.4, we have the following

$$\|M_\ell^g(\lambda, \mathbf{u}) - M_\ell^g(\widehat{\lambda}, \widehat{\mathbf{u}}) - M_\ell^g(\lambda_{[k_{\mathsf{res}}]}, \mathbf{u}_{[k_{\mathsf{res}}]})\|_2 \leq \xi' \tag{11}$$

for all $\ell = 2, 4, \ldots, 2k+2$. By Lemma 4.2, the deviation between the empirical mean $\frac{1}{N}\sum_{a=1}^N (y_a - f_{\widehat{\lambda},\widehat{\mathbf{u}}}(x_a))x_a$ and the population mean is bounded by $\xi'/2$ provided $N \geq d^{O(\ell)}\,(\mathcal{R}^2 + \|\widehat{\lambda}\|_1^2)/\xi'^2$. This establishes the second bound by triangle inequality with (11).

B.3 Proof of Lemma 4.5

By AM-GM, it suffices to show

$$\|f_{w,\lambda,\mathbf{u}} - f_{0,\widehat{\lambda},\widehat{\mathbf{u}}} - h\| \lesssim \mathrm{poly}(d, \mathcal{R})(\varepsilon' + \xi')/\omega + \omega \tag{12}$$

holds for any $0 < \omega < 1$. By hypothesis, for all $i, j \in [k_{\mathsf{res}}]$ we have that $|v_i - v_j| \leq \varepsilon'$, so by the fact that we are conditioning on the event of Lemma 2.2,

$$\|u_i - u_j\| \lesssim \varepsilon' k^2 \sqrt{d}\,.$$

Let S^+ and S^- denote the partition of $[k_{\mathsf{res}}]$ into indices i for which $\mathrm{argmin}_{\sigma\in\{\pm 1\}}\|u_i - \sigma \cdot u_1\|$ is $+1$ and -1 respectively. Also define $\lambda^+ \triangleq \sum_{i\in S^+}\lambda_i$ and $\lambda^- \triangleq \sum_{i\in S^-}\lambda_i$. Consider the network

$$h^* \triangleq \lambda^+ \mathsf{relu}(\langle u_1, x\rangle) + \lambda^- \mathsf{relu}(\langle -u_1, x\rangle)\,.$$

Then by Lemma A.2,

$$\|h^* - f_{\mathsf{res}}\| \lesssim \varepsilon' \mathcal{R}\, k^2 \sqrt{d}\,. \tag{13}$$

By Lemma A.3 applied to $h^* - f_{\mathsf{res}}$, we conclude that

$$\|T_1((\lambda^+, \lambda^-), (u_1, -u_1)) - T_1(\lambda_{[k_{\mathsf{res}}]}, \mathbf{u}_{[k_{\mathsf{res}}]})\|_2 \lesssim \varepsilon' \mathcal{R}\, k^2 d^{3/2}$$
$$\|T_2((\lambda^+, \lambda^-), (u_1, -u_1)) - T_2(\lambda_{[k_{\mathsf{res}}]}, \mathbf{u}_{[k_{\mathsf{res}}]})\|_F \lesssim \varepsilon' \mathcal{R}\, k^2 d^2 \varepsilon'\,.$$

By combining these with Lemma 4.3, we find that the empirical estimates $\frac{1}{N}\sum_{a=1}^{N} 2(y_a - f_{\widehat{\lambda},\widehat{\mathbf{u}}}(x_a))x_a$ and $\frac{1}{N}\sum_{a=1}^{N}\sqrt{2\pi}(y_a - f_{\widehat{\lambda},\widehat{\mathbf{u}}}(x_a))(x_a x_a^\top - \mathrm{Id})$ are $(\xi' + \mathrm{poly}(d)\mathcal{R}\varepsilon')$-close to

$$T_1((\lambda^+,\lambda^-),(u_1,-u_1)) = (\lambda^+ - \lambda^-)u \text{ and } T_2((\lambda^+,\lambda^-),(u_1,-u_1)) = (\lambda^+ + \lambda^-)u_1 u_1^\top,$$

provided $N \geq \mathrm{poly}(d)\,(\mathcal{R}^2 + \|\widehat{\lambda}\|_1^2)/\xi'^2$. By right-multiplying the latter empirical estimate by g, we get an estimate of $T_2((\lambda^+,\lambda^-),(u_1,-u_1))\,g = \langle u_1, g\rangle \cdot (\lambda^+ + \lambda^-)\cdot u$ whose error in L_2 is of the same order. By the fact that we are conditioning on the event of Lemma 2.3, $|\langle u_1, g\rangle| \gtrsim 1/k\sqrt{d}$.

We conclude that we have access to both $(\lambda^+ + \lambda^-)u_1$ and $(\lambda^+ - \lambda^-)u_1$, and thus to $\lambda^+ u_1$ and $\lambda^- u_1$ and also the scalars λ^+ and λ^-, to error of order $\mathrm{poly}(d)(\mathcal{R}\varepsilon' + \xi')$. If λ^+ and λ^- are both at most $c\omega$ in magnitude for sufficiently small c, then $\|h^*\| \lesssim \omega$ and the estimator $h \equiv 0$ already achieves the desired bound in (12). Otherwise, we can use our estimates of $\lambda^+ u_1$ and $\lambda^- u_1$ to estimate u_1 to L_2 error of order $\mathrm{poly}(d)(\mathcal{R}\varepsilon' + \xi)/\omega$. By Lemma A.2, we obtain an estimate $h = \mu^+\mathsf{relu}(\langle u, \cdot\rangle) + \mu^-\mathsf{relu}(\langle -u, \cdot\rangle)$ satisfying

$$\|h^* - h\| \leq \mathrm{poly}(d,\mathcal{R})(\varepsilon' + \xi')/\omega\,. \tag{14}$$

Combining Eqs. (2), (13), (14) yields the desired bound (12) upon noting that the bound on $\|h^* - h\|$ dominates among the three bounds.

B.4 Proof of Lemma 4.17

Since in a one-hidden-layer ReLU network (without bias term), $F(0) = 0$, and furthermore F is a $2k\mathcal{R}$-Lipschitz continuous function, we conclude that $|F(x)| \leqslant 2k\mathcal{R}\cdot\|x\|_2$ for $\forall x \in \mathbb{R}^d$. Next, we can apply the proof of Lemma A.1 of [CKM21] to show that $G(x) := F(x)^2 - \mu$ is a zero-centered, sub-exponential random variable with sub-exponential norm $\|G\|_{\Psi_1} = \mathcal{O}(\mu + 4\mathcal{R}^2k^3)$. Finally, by using the concentration property of sub-exponential random variables, we conclude that:

$$\left|\frac{1}{N}\sum_{i=1}^{N} G(x_i)\right| \leqslant t$$

with probability at least $1-\delta$. Here, the sample size $N = \Theta(K^2\log(1/\delta)/t^2)$, where $K = \mu + 4\mathcal{R}^2k^3$.