CS 2243 Fall 2024 Scribes: <names>
10/21 Based on notes by: Cole Harten, Weiyuan Gong

Lecture 13: Hardness Basics, SQ Lower Bounds, SQ Dimension

Today, we introduce the next unit on computational complexity, specifically on proving lower bounds of problems. These lower bounds serve to tell us when continuing to find a more efficient algorithm for a problem is futile and when we should revist and possibly alter the assumptions we made.

1 Finishing Lecture 12

1.1 Finishing Mean Field Limit

Recall that last time we hade a MLP with scaling defined as

$$f_\theta(x) = \frac{1}{N}\sum_i a_i \sigma\left(\langle w_i, x\rangle\right) \tag{1}$$

Here N represents the "width" of the network (the number of neurons). We have previously observed that as $N \longrightarrow \infty$, the neurons at time t of the gradient flow will converge to i.i.d. draws from the distribution ρ_t, which satisfies the differential equation

$$\partial_t \rho_t = \operatorname{div}\left(\rho_t \cdot \nabla \psi_{\rho_t}\right) \tag{2}$$

However, in many settings, this PDE is intractable to work with. Hence, in order to greatly simplify the problem, we often consider toy models, specifically ones with high symmetry. For example, one such simplified, model that makes use of symmetry is when the x' s are normally distributed and $y = \phi\left(\langle w^*, x\rangle\right)$. This is a type of single index model, where the data depends only on a fixed projection of the data (in the direction of w^*). Hence, in this model the data is secretly 1-dimensional. As in this situation, the data distribution doesn't vary with rotations so long as w^* is preserved, this greatly simplifies the PDE.

In these settings you can often numerically solve the PDE to obtain actual predictions. However, such numerical methods rely on the simplifying assumptions we made earlier. However, it is still difficult to get provable results from these algorithms.

Now we will revisit the correlational statistical query model (CSQ). For single index-models (models of the form $y = \phi\left(\langle w^*, x\rangle\right)$), the complexity of CSQ algorithms are dictated primarily by what is known as the "informal exponent," that is, the smallest s for which the s-th Hermite coefficient of ϕ is non-zero. If we are training only a single neuron, running online SGD will learn in an expected amount of time, $O\left(d^s\right)$ [AGJ21]. There are results that generalize the idea of "leap complexity" to multi-index models (models of the form $y = \phi\left(\prod_W x\right)$) in time $O\left(d^{\text{leap}}\right)$ [ABAM23].

In general, the algorithms discussed above are optimal for CSQ algorithms. However, there are (at least in principle) other more efficient non-CSQ algorithms, such as filtered PCA which achieve O(d) sample complexity and fixed-polynomial runtime. The optimality of these CSQ algorithms, can be proved using computational lowerbounds (which we will discuss in a moment).

1.2 Recap of Supervised Learning

Still virtually all algorithms in the field of PAC learning, rely on low-degree polynomials. We began with approximating binary circuits by their low degree Fourier coefficients and ended with using the Mean Field Limit to learn low-degree components of the underlying functions generating the data.

However, there are two distinct concepts of low degree polynomials have discussed. The first is What is the smallest degree for which there is a polynomial that approximates the ground truth? This approach exploits Fourier/Hermite concentrations. Algorithms include low-degree algorithm, polynomial regression, and kernel methods.

The other concept of low-degree polynomials is What is the smallest degree at which the ground truth has a nonzero polynomial component? This approach exploits informa-
tion exponents/leaps. Algorithms included tensor methods, feature learning/GD beyond NTK.

2 Introduction to Computational Complexity

2.1 Guiding Examples

Throughout the entire course, we have consistently seen two examples show up:

- Learning a mixture of k Gaussians in $\mathbb{R}^d$
- Learning neural networks of size k over Gaussian inputs in $\mathbb{R}^d$

For both algorithms, it is an open question regarding the existence of a fully poly (k, d)-time algorithm. This could either be because we just haven't found some mathematical solution yet, or it could be because no efficient algorithm exists as the problem is currently formulated. To rule out the first option, we make use of lower bounds.

2.2 Computational Hardness

We have dealt with several problems restricted by statistical lower bounds. For example, the Airy Disks from the first lecture will require exponentially many samples to differentiate below the diffraction limit. In this case, solving this problem without enough data is impossible even with an infinite amount of compute.

In this section, we will explore the case where there is enough signal in the dataset that a computationally inefficient algorithm (i.e. brute force) can solve the problem, but that no computationally efficient algorithm exists to solve the problem. There are a couple of different version of classical computational hardness such as NP hardness (SAT, graph coloring, etc.) and cartographic hardness (public-key cryptography, pseudorandom generators, etc.).

However, we will focus primarily on "average case hardness." A classic example is the planted clique problem:
Setup: Given the adjacency matrix of a graph sampled either from an Erdos-Renyi graph $G(n, 1/2)$ (every edge independently included with probability 1/2) or a Planted graph (graph is $G(n, 1/2)$ with a random clique of size N).
Task: Determine which case we are in with high probability over the randomness
of the instance.
The largest clique in $G(n, 1/2)$ is of size either $2 \log n$ or $2 \log n + 1$ with probability $1 - o_n(1)$. Thus, the problem is information-theoretically intractable when $N \in \Omega(\log n)$. We could brute force a solution to this algorithm (i.e. there is enough "signal"), but no known computationally efficient algorithm exists for $N = o(\sqrt{N})$. In the case of $N \in \Omega(\sqrt{N})$, there does exist an efficient algorithm using the top eigen vector of the adjacency matrix [AKS98]. It is hypothesized that no poly-time algorithm exists for this case.

Another problem is learning parity with noise, which is a noisy supervised learning task. For positive η, random $S \subseteq [d]$, and dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, defined as

$$x \sim \{\pm 1\}^d \quad y = \begin{cases} x_S & w \cdot p.1 - \eta \\ -x_S & \text{otherwise} \end{cases}$$

$N = O(d \log d)$ samples are information-theoretically sufficient, as we could (inefficiently) brute force the solution. When we have $\eta = 0$, this is just a linear system modulo 2 that we could solve in polynomial time with Gaussian elimination. It is hypothesized that no polynomial time algorithm exists even to distinguish the y' s from random labels.

2.3 Traditional Hardness Paradigm

The classic approach to proving hardness rests of reductions. Given some problem X that is provably hard, show that efficiently solving some other problem Y could be (efficiently) mapped to a solution of problem X. This approach is very effective for worst-case hardness but break down for average-case hardness. This requires us to look for new ways to prove average-case hardness that do not rely on reductions.

3 Lower Bounds in Restricted Models

3.1 Restricted Model of Computation

Here is an alternate approach to proving these lower-bounds:

- Formally define a restricted model of computation that captures all known algorithms for the problem in question.
- Prove a lower bound against algorithms in the restricted model

This approach works unconditionally (unlike NP hardness which is conditional on $P \neq NP$). There are a few different approaches to this proof:

- Statistical query: Only makes use of noisy estimates of population-level statistics of the data distribution
- Lipschitz algorithms: If the instance is pertubed, the algorithm output does not change much.
- Low-degree algorithms: Only uses low-degree polynomials evaluated on the data
- Sum-of-Squares Algorithms: There is a "Canonical" SoS relaxation of the problem, and we want to prove high degree SoS is necessary

These are much "easier" to show than reductions; however, they are often less convincing than reduction-based bounds.

3.2 Statistical Query

We will return to the correlational statistical query (CSQ) model of computation. We will define the statistical query (SQ) model of computation as an algorithm that only interacts with some dataset $\{(x_i, y_i)\}_{i=1}^N$ through some oracle that takes in

$$\psi : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$$

and outputs

$$\mathbb{E}[\psi(x, y)] + \text{ noise}$$

where $|\text{ noise }| \leq \tau$ for some tolerance $\tau = \sqrt{1/N}$.
If the labels $y_i \in \{\pm 1\}$ and $x \sim q$ for some unknown distribution q, then $SQ = CSQ$ and we have

$$\mathbb{E}[\psi(x, y)] = \mathbb{E}\left[\frac{1+y}{2}\psi(x, 1) + \frac{1-y}{2}\psi(x, -1)\right] = \mathbb{E}[g(x)] + \mathbb{E}[y \cdot h(x)] \tag{3}$$

In general, the statistical query model can capture essentially any known learning algorithm except for Gaussian elimination and others (it is still an open question which algorithms it cannot capture).

3.3 Proof of SQ Lower Bound for Noiseless Parity

Theorem 1 ([Kea98]). *Any statistical query algorithm for learning parity (without noise) requires* $2^{\Omega(d)}$ *queries or tolerance* $2^{-\Omega(d)}$.

Proof. Recall the setting from above for learning parity with noise. We have the data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ with

$$x \sim \{\pm 1\}^d \quad y = \begin{cases} x_S & w \cdot p \cdot 1 - \eta \\ -x_S & \text{otherwise} \end{cases}$$

Now, however, for the setting without noise, we will set $\eta = 0$. Consider first any CSQ $\phi : \{\pm 1\}^d \to [-1, 1]$. Let

$$\phi_S := \mathbb{E}_x \left[x_S \phi(x)\right]$$

I claim first that for uniformly random subsets $S \subseteq [n]$,

$$\mathrm{Var}_S \left[\phi_S\right] \geq 2^{-\Omega(n)}$$

The proof of this lemma is as follows:

$$\begin{aligned} \mathrm{Var}\left[\phi_S\right] &= \mathop{\mathbb{E}}_{S}\left[\phi_S^2\right] - \mathop{\mathbb{E}}_{S}\left[\phi_S\right]^2 \\ &= \mathop{\mathbb{E}}_{S}\mathop{\mathbb{E}}_{x,x'}\left[x_S x'_S \phi(x)\phi\left(x'\right)\right] - \mathop{\mathbb{E}}_{S,S'}\mathop{\mathbb{E}}_{x,x'}\left[x_S x'_{S'} \phi(x)\phi\left(x'\right)\right]\Big] \\ &= \mathop{\mathbb{E}}_{x,x'}\left[\phi(x)\phi\left(x'\right) \underbrace{\mathop{\mathbb{E}}_{S,S'}\left[x_S x'_S - x_S x - S'\right]}_{\alpha}\right] \quad (*) \end{aligned} \tag{*}$$

I claim that the inside expectation α is equal to 0 if $x \neq x'$. For $z \neq 1, E_S\left[z_S\right] = 0$ so if $x \neq x'$, then $\mathbb{E}_S\left[x_S x'_S\right] = \mathbb{E}\left[z_S\right] = 0$ and $\mathbb{E}_{S,S'}\left[x_S x'_S\right] = \mathbb{E}, \left[x_S\right]\mathbb{E}_{S'}\left[x'_{S'}\right] = 0$ because at most of of $x, x' = 1$. This is a "pairwise independence" argument.

Returning, to the previous expression, we can use this fact to get

$$\begin{aligned} (*) &= \mathop{\mathbb{E}}_{x,x'}\left[\phi(x)\phi\left(x'\right) \cdot \mathbb{1}\left[x = x'\right]\right] \\ &= \frac{1}{2^n}\mathop{\mathbb{E}}_{x}\left[\phi(x)^2\right] \\ &\leq \frac{1}{2^n}. \end{aligned}$$

Thus, by Chebyshev's inequality, we can get that

$$\begin{aligned} \mathop{\mathbb{P}}_{S}\left[\left|\phi_S - \mathop{\mathbb{E}}_{S}\left[\phi_s\right]\right| \geq \tau\right] &\leq \frac{1}{\tau^2}\mathrm{Var}\left(\phi_S\right) \\ &\leq \frac{1}{2^n \tau^2} \end{aligned}$$

Hence, in order to answer the CSQ ϕ, we can just output $\mathbb{E}_S\left[\phi_S\right]$. If the tolerance is equal to τ, this is accurate for $\frac{1}{2^n\tau^2}$ fraction of parity functions. That is, each CSQ only rules out at most $1/\tau^2$ many S' s. Thus, we need $2^n\tau^2$ many queries. Letting $\tau := 2^{-n/3}$ completes the proof.

3.4 SQ Dimension

Now, we provide a gneral recipe for proving CSQ lower bounds for supervised problems.

Definition 1 (SQ dimension). *A class of functions has SQ dimension $\geq D$ w.r.t. input distribution q if there exist functions $f_1, \ldots, f_D$ in the class s.t. for all $i \neq j$:*

$$|\mathbb{E}_{x \sim q}[f_i(x) f_j(x)]| \leq \frac{1}{D}.$$

As an example, we consider the PARITY $= \{f(x) = x_S : S \subset [d]\}$ has SQ dimension 2^d with respect to uniform distribution. This argument also lets us generalize to sparse parity.

Now, we give the following theorem connecting the SQ dimension and CSQ query lower bounds.

Theorem 2 ([BFJ+94, Szö09]). *If $\mathcal{F}$ has SQ dimension D with respect to q, then any CSQ algorithm for learning $\mathcal{F}$ from examples from q requires $\Omega(D\tau^2)$ queries or tolerance τ.*

As an example, we take the tolerance to be $\tau = \sqrt[3]{1/D}$,. This indicates that If D is super-polynomially large, the SQ lower bound qualitatively implies that you either need a super-polynomial number of queries or a super-polynomial number of samples (inverse tolerance).

proof on board. We consider $\mathcal{F}$ with SQ dimension $\geq D$, i.e., we can find $f_1, \ldots, f_D \in \mathcal{F}$ such that

$$|\mathbb{E}_x[f_i(x) f_j(x)]| \leq \frac{1}{D}.$$

The brief idea for the proof is to argue that the number of problem instances $i \in [D]$ that get "ruled out" at every state is very small because for most i, answering with a "trivial" oracle response is accurate.

In particular, we define

$$\langle f, g \rangle := \mathbb{E}_{x \sim q}[f(x) g(x)].$$

Fixing CSQ query $\mathbb{E}[y\psi(x)]$, we further define

$$A^+ := \{i \in [D] : \langle f_i, \psi \rangle \geq \tau\},$$
$$A^- := \{i \in [D] : \langle f_i, \psi \rangle \leq -\tau\}.$$

Our goal is to show that $|A^{\pm}|$ are small. We pick the rotation function

$$Z = \langle \psi, \sum_{i \in A^+} f_i \rangle^2.$$

We first provide the upper bound of Z by Cauchy-Schwartz inequality:

$$\begin{aligned}
Z &\leq \|\psi\|^2 \| \sum_{i \in A^+ f_i} \|^2 \\
&\leq \sum_{i,j \in A^+} \langle f_i, f_j \rangle \\
&= \sum_{i \in A^+} \|f_i\|^2 + \sum_{i \neq j \in A^+} \langle f_i, f_j \rangle \\
&\leq \sum_{i \in A^+} \|f_i\|^2 + \frac{1}{D} |A^+|(|A^+| - 1) \\
&\leq \frac{|A^+|^2}{D} + |A^+|.
\end{aligned}$$

We also have the following lower bound on Z according to the fat that $\langle \psi, \sum_{i \in A^+} f_i \rangle \geq \tau |A^+|$. By definition of A^+, we have

$$Z \geq \tau^2 |A^+|^2.$$

Therefore, we have

$$\tau^2|A^+|^2 \leq Z \leq \frac{|A^+|^2}{D} + |A^+|,$$

which indicates that

$$|A^+| \leq \frac{D}{D\tau^2 - 1} \leq O(1/\tau^2).$$

Similarly, we have $|A^-| \leq O(1/\tau^2)$. This shows that all but $O(1/\tau^2)$ are consistent with the answer 0. Therefore, $D\tau^2$ queries are enough to narrow down to the true answer. □

References

[ABAM23] Emmanuel Abbe, Enric Boix-Adsera, and Theodor Misiakiewicz. Sgd learning on neural networks: leap complexity and saddle-to-saddle dynamics, 2023.

[AGJ21] Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. Online stochastic gradient descent on non-convex losses from high-dimensional inference, 2021.

[AKS98] Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. *Random Structures & Algorithms*, 13(3-4):457–466, 1998.

[BFJ$^+$94] Avrim Blum, Merrick Furst, Jeffrey Jackson, Michael Kearns, Yishay Mansour, and Steven Rudich. Weakly learning dnf and characterizing statistical query learning using fourier analysis. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 253–262, 1994.

[Kea98] Michael Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM (JACM)*, 45(6):983–1006, 1998.

[Szö09] Balázs Szörényi. Characterizing statistical query learning: simplified notions and proofs. In *International Conference on Algorithmic Learning Theory*, pages 186–200. Springer, 2009.