CS 839: Theoretical Foundations of Deep Learning		Spring 2023
Lecture 20 Lower Bounds in Statistical Query model		
Instructor: Yingyu Liang	*Date:*	*Scriber: Cheuk Yin (Eric) Lin*

In this lecture, we introduce the statistical query (SQ) model and the lower bound of learning neural networks using the statistical query dimension. The purpose of this lecture is to introduce the SQ lower bound technique, a tool often used in deriving complexity results and is more specific for machine learning. All the lower bounds essentially depend on certain computational models, where typical lower bound bounds base on the NP $\neq$ P hardness assumption. SQ lower bound is another kind of lower bound which does not assume that certain family of problems is hard. Instead SQ lower bound considers the statistical query algorithms, which essentially covers all the learning algorithms we know (except for Gaussian elimination).

1 Statistical Query Model

We first define the SQ model formally. We assume that the learning algorithm can only receive information about the data through statistical queries. A statistical query is specified by some polynomially-computable property predicate Q of labeled instances and a tolerance parameter $\tau \in [0, 1]$ over $(x, y) \sim \mathcal{D}$ where $\mathcal{D}$ is the data distribution. For a query (Q, τ), the algorithm receives a response $\widehat{P}_Q \in [P_Q - \tau, P_Q + \tau]$, where $P_Q = \Pr[Q(x, y) \text{ is true}]$.

2 Lower bounds for learning without input structure

Definition 1 (Definition 2 in [1])**.** For concept class C and distribution $\mathcal{D}$, the statistical query dimension *SQ-DIM*$(C, \mathcal{D})$ is the largest number d such that C contains d concepts $c_1, ..., c_d$ that are nearly pairwise uncorrelated: specifically, for all $i \neq j$,

$$\left| \Pr_{x\sim\mathcal{D}}[c_i(x) = c_j(x)] - \Pr_{x\sim\mathcal{D}}[c_i(x) \neq c_j(x)] \right| \leq 1/d^3.$$

Theorem 2. (Theorem 12 in [1]) In order to learn C to error less than $1/2 - 1/d^3$ in the SQ model, where $d =$ *SQ-DIM*$(C, \mathcal{D})$, either the number of queries or $1/\tau$ must be at least $d^3/2$.

To show that learning is hard, we want to show that the SQ dimension for some concept class that can be approximated our learned function is not small. We can consider some hard concept classes and show that their SQ dimension is large, thus we show a lower bound on network learning. In particular, we consider the family of parity functions, which is known as the most uncorrelated family of function. Let us first give the formal definition of parity functions.

Let $\mathcal{X} = \{\pm 1\}^d \sim \mathcal{D}$ where $\mathcal{D}$ uniform. For any $A \in [D]$, we define the concept class $\mathcal{C} : y = g_A(x) = \prod_{j\in A} x_j$ where $|A| = k$. It is said to be the most uncorrelated family of

function because for any $A, B \subseteq [D]$, $A \neq B$, g_A and g_B are orthogonal in expectation within the functional space. In particular we have

$$\begin{aligned}
\boldsymbol{E}_{x\sim\mathcal{D}}^{A\neq B}[g_A(x)g_B(x)] &= \boldsymbol{E}\left[\prod_{j\in A\cap B} x_j^2 \prod_{j\in A\setminus B} x_j \prod_{j\in B\setminus A} x_j\right] \\
&= \boldsymbol{E}\left[\prod_{j\in A\setminus B} x_j\right]\boldsymbol{E}\left[\prod_{j\in A\cap B} x_j^2 \prod_{j\in B\setminus A} x_j\right] \quad \text{w.l.o.g. that } A\setminus B \text{ non-empty} \\
&= 0.
\end{aligned}$$

This feature of parity functions implies $d = \textit{SQ-DIM}(\mathcal{C}, \mathcal{D}) = \binom{D}{k}$, and hence we immediately get a exponential in k lower bound on the number of queries to learn the above concept class.

3 Approximating Parity Functions with 2-Layer Neural Networks

We now show that there is a two-layer network that can fit the labels generated by any parity function, which is the "hard" concept class described above.

Lemma 3. For any parity function g_A where $A \subseteq [D]$ and $|A| = k$, there exists a 2-layer neural network $f(x)$ with ReLU activation functions s.t. $f(x) = g_A(x)$ for all $x \in \mathcal{X}$ and f has $\Theta(k)$ hidden units.

Proof. We show how to construct a 2-layer neural network with $\Theta(k)$ that can perfectly represent g_A for any $A \subseteq [D]$. Consider the case when k is even (note that the proof for k odd is similar, so we omit it here).

We define $z_j = (x_j + 1)/2 \in \{0, 1\}$, mapping x_j from $\{-1, +1\}$ to $\{0, 1\}$. Denote $z = \sum_{j\in A} z_j$ the count of ones in subset A. Therefore if z is even, then $g_A(x) = +1$, and if z is odd, then $g_A(x) = -1$. Notice that $z \in \{0, ..., k\}$, so the idea is to use indicator function to output each possible value of z. If we can get the function $\delta_s(z)$ as illustrated in Diagram 3 with ReLUs, then we can build a 2-layer neural network as follows:

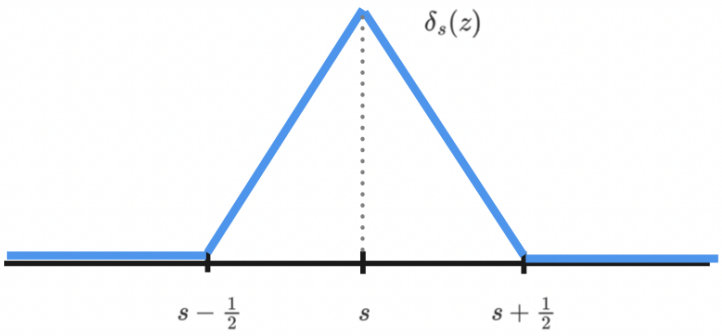
$$f(x) = \sum_{s\in[k]} \delta_s(z)(-1)^{k-s}.$$

Let $\sigma_1(z) = \sigma(z - (s - 1/2))$, $\sigma_2(z) = \sigma(z - s)$ and $\sigma_3(z) = \sigma(z - (s + 1/2))$ where $\sigma(z)$ is the standard ReLU activation. Then for any $s \in \mathbb{R}$, we can write

$$\delta_s(z) = 2(\sigma_1 - \sigma_2) - 2(\sigma_2 - \sigma_3) = 2\sigma_1 - 4\sigma_2 + 2\sigma_3$$

and this is simply a linear combination of three ReLU nodes. Consequentially we can construct a 2-layer neural network with ReLU activation to perfectly fit the labels generated by g_A for any $A \subseteq [D]$ with at most $3(k + 1) = \Theta(k)$ hidden units.

□



If you can consider learning a concept class of a 2-layer neural network with $\Theta(k)$ hidden units, then it covers the family of parity functions as a special case. This lemma tells us that learning neural networks in general should be quite hard.

4 Other remarks

- Instead of showing the worst case learning of the concept class, we could also consider the average case learning guarantee by assuming a certain distribution over our concept class. However, recent results have shown that learning neural networks in this weaker case still hard (not quite NP-hard, but quite close in terms of hardness). This tells us that we need to impose additional assumptions such as the input distribution in order to better guarantees.

- For example, consider a special input distribution as follows: Suppose we generate labels with $y = g_A(x)$ for some $A \subseteq [D]$, and x follows that $x_A = [1, ..., 1]^T$ with probability $2/3$ and $x_A = [-1, ..., -1]^T$ with probability $1/3$, $x_{\neg A} = \text{Unif}(\pm 1)$. Since the input distribution heavily depends on A and correlates with the ground truth concept, we can see that this concept can be easily learned with a constant number of samples by summing up the values of samples at each coordinate to determine whether it is part of A.

References

[1] Avrim Blum, Merrick Furst, Jeffrey Jackson, Michael Kearns, Yishay Mansour, and Steven Rudich. Weakly learning dnf and characterizing statistical query learning using fourier analysis. *Proceedings of the Twenty-Sixth Annual Symposium on Theory of Computing*, 10 1994.