CSCI699: Machine Learning Theory Fall 2023

Lecture 7: SQ Learnability, Computational-Statistical Tradeoffs

Instructor: Vatsal Sharan

These lecture notes are based on scribe notes by Neel Patel and Ali Omrani.

1 SQ Learnability and SQ Dimension

Last time we defined uncorrelated functions, and SQ dimension.

Definition 1. *Two functions f, g defined on the same domain are uncorrelated if*

$$\Pr_{x \sim D}[f(x) = g(x)].$$

Definition 2. *The SQ-dimension $\mathsf{SQDIM_D}(\mathcal{C})$ of a class $\mathcal{C}$ w.r.t. a distribution D over $\mathcal{X}$ is the size of the largest subset $\mathcal{C}' \subset \mathcal{C}$ such that for all $f, g \in \mathcal{C}'$*

$$\left| \Pr_{x \sim D}[f(x) = g(x)] - 1/2 \right| < 1/|C'|.$$

The SQ dimension of a concept class is large if there are a large number of concepts which are very different from each other (similar to how VC dimension is large if it is possible to shatter a dataset with the functions in the class). We mentioned that SQ dimension closely captures learnability in the SQ model. Similar to how VC dimension gave a measure of complexity for a hypothesis class which characterizes how much data is needed for learning, SQ dimension captures when *efficient algorithms* maybe able to learn the concept, at least within the class of algorithms which work in the SQ framework. We discussed how most algorithms (such as gradient descent and its variants) can in fact be implemented in the SQ model. Therefore the SQ model and SQ dimension can be a useful tool for understanding when a certain problem can be solved computationally efficiently.

This lecture, we will prove upper and lower bounds for learnability using SQ dimension.

2 Weak learnability from small SQ dimension

Let us begin with the notion of *weak learning* last time (we will explore the relationship between weak learning usual PAC learning in a few lectures):

Definition 3 (Weak Learning)**.** *An algorithm A is a weak learner with advantage γ for class $\mathcal{C}$ if: for any dist. D and any target $c \in \mathcal{C}$, given access to $EX(c, D)$, w.p. $(1-\delta)$, produces a hypotheses with error$(h; c, D) \leq \gamma$.*

The following theorem shows that SQ dimension captures weak learnability.

Theorem 4. *If $\mathsf{SQDIM_D}(\mathcal{C}) = \mathsf{poly(d)}$, then we can efficiently "weakly learn" $\mathcal{C}$ over D (get error rate $\leq 1/2 - 1/poly(d)$) using a SQ-learning algorithm.*

Proof. Let $s = \mathsf{SQDIM_D}(\mathcal{C})$, let $\mathcal{H} \subseteq \mathcal{C}$ be maximal subset such that $\forall h_i, h_j \in \mathcal{H}$, we have

$$\left| \Pr_D[h_i(x) = h_j(x)] - \frac{1}{2} \right| < \frac{1}{s+1}.$$

Note that we must have $|\mathcal{H}| \leq s$. Otherwise the SQ dimension of $\mathcal{H}$ would be $s+1$, which would be a contradiction since the SQ dimension is s.

Given this set $\mathcal{H}$, we try every $h_i \in \mathcal{H}$ and use the SQ-oracle to estimate its error. Note that since $s = \text{poly}(d)$, the number of queries made is still polynomial in d and hence this algorithm is an efficient SQ algorithm.

We claim that at least one h_i or $\bar{h}_i$ (the complement or negation of h_i) must be a weak leaner. To verify this note that if the target c satisfied:

$$\left| \Pr_D[h_i(x) = c(x)] - \frac{1}{2} \right| < \frac{1}{s+1} \quad \forall h_i \in \mathcal{H}$$

then we can include c in the set $\mathcal{H}$, which is a contradiction since $\mathcal{H}$ is a maximal set. Therefore, at least one $h_i \in \mathcal{H}$ must satisfy,

$$\left| \Pr_D[h_i(x) = c(x)] - \frac{1}{2} \right| \geq \frac{1}{s+1}$$

This implies that either h_i or $\bar{h}_i$ must get error at most $1/2 - 1/(s+1)$. Since $s = \text{poly}(d)$, there exists at least one weak learner in $\mathcal{H}$. ■

3 Parities, and impossibility of SQ learning

The previous result shows that small SQ dimension is sufficient for (weak) learning. We now consider the other direction, and show that if the SQ dimension is large then (weak) learning is impossible.

Theorem 5. *If a concept class $\mathcal{C}$ has $\mathsf{SQDIM_D}(\mathcal{C}) = \mathsf{s}$, then any SQ algorithm for learning $\mathcal{C}$ over the distribution D which makes queries of tolerance $\tau > \tau_{min}$ must make $\Omega(\tau_{min}^2 s)$ queries to the SQ oracle to learn $\mathcal{C}$. Therefore, if $\mathsf{SQDIM_D}(\mathcal{C}) > \mathsf{poly(d)}$ then it is not possible to efficiently learn $\mathcal{C}$ over D using SQ-algorithms (even "weak-learning" to error $= 1/2 - 1/\text{poly}(d)$ is impossible).*

An example of a hypothesis class with large SQ dimension is PARITIES, the class of parity functions. We will see PARITIES quite a bit this lecture, and they appear to be quite fundamental to understand computational hardness of learning. We first define the class of parity functions,

$$\begin{aligned} \mathcal{X}^d &= \{0,1\}^d \\ \mathcal{Y} &= \{0,1\} \\ \mathcal{C} &= \{w(x) = \langle w, x \rangle \mod 2 : w \in \{0,1\}^d\}. \end{aligned}$$

We first show that this concept class is efficiently PAC learnable.

Theorem 6. *PARITIES are efficiently PAC learnable.*

Proof. Note that since $|C| = 2^d$, ERM gets error ϵ with $O\left(\frac{d}{\epsilon}\log\left(\frac{1}{\delta}\right)\right)$ samples with probability $1-\delta$.

To finish the argument, we need to show that we can solve the ERM problem in polynomial time. Suppose we get examples $\{(x_1, b_1), (x_2, b_2), \dots, (x_n, b_n)\}$. We can solve the following linear system to find w_{ERM}, which gets zero training error and hence is an ERM.

$$\begin{pmatrix} \text{——}x_1\text{——} \\ \text{——}x_2\text{——} \\ \cdot \\ \cdot \\ \cdot \\ \text{——}x_n\text{——} \end{pmatrix} \begin{pmatrix} \\ \\ w_{ERM} \\ \\ \\ \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ \cdot \\ b_n \end{pmatrix} \bmod 2$$

This is a linear system over $\mathbb{F}_2$ (modulo 2), and standard algorithms such as Gaussian elimination can solve it in polynomial time. ■

Next we will show that this concept class cannot be learned in the SQ model over the uniform distribution, because it has exponentially large SQ dimension.

It will be convenient to define PARITIES as a function on $\{-1, +1\}^d \mapsto \{-1, +1\}$

$$\mathcal{X}^d = \{\pm 1\}^d$$
$$\mathcal{Y} = \{\pm 1\}$$
$$\mathcal{C} = \{c_S(x) = \Pi_{i \in S} x_i : S \subseteq \{1, \dots d\}\}$$

Let U be the uniform distribution over $\{\pm 1\}^d$.

Lemma 7. *If $S \neq T$, $\Pr_{x \sim U}(c_S(x) = c_T(x)) = 1/2$.*

Proof. Let $S \Delta T = \{S - T\} \cup \{T - S\}$. Note that

$$\begin{aligned} \mathop{\mathbb{E}}_{x \sim U}[c_S(x) \cdot c_T(x)] &= \mathop{\mathbb{E}}_{x \sim U}\left[\prod_{i \in S} x_i \cdot \prod_{i \in T} x_i\right] \\ &= \mathop{\mathbb{E}}_{x \sim U}\left[\prod_{i \in S \Delta T} x_i\right] \\ &= 0 \text{ if } S \neq T \end{aligned}$$

where the last step follows since each x_i is uniformly distributed on $\{\pm 1\}$. Let us verify that this is sufficient. Note that

$$\Pr_{x \sim U}(c_S(x) = c_T(x)) + \Pr_{x \sim U}(c_S(x) \neq c_T(x)) = 1$$

Therefore since

$$\mathop{\mathbb{E}}_{x \sim U}[c_S(x) \cdot c_T(x)] = \Pr_{x \sim U}(c_S(x) = c_T(x)) - \Pr_{x \sim U}(c_S(x) \neq c_T(x)),$$

$\mathop{\mathbb{E}}_{x\sim U}[c_S(x)\cdot c_T(x)] = 0$ implies $\Pr_{x\sim U}(c_S(x) = c_T(x)) = 1/2$.

■

By Lemma 7, $\mathsf{SQDIM}_\mathsf{U}(\mathcal{C}) = 2^\mathsf{d}$. Therefore we have the following result about impossibility of learning parities in the SQ model.

Corollary 8 (Corollary of Theorem 5)**.** *It is not possible to efficiently learn parities over the uniform distribution in the SQ model.*

We will now prove Theorem 5 for the special case of PARITIES.

Theorem 9 (Hardness of parities in SQ)**.** *Any SQ algorithm for learning PARITIES over $D = U$ which makes queries of tolerance $\tau > \tau_{min}$ must make $\Omega(\tau_{min}^2 2^d)$ queries to the SQ oracle.*

Proof. Recall the definition of the Correlational SQ (CSQ) oracle, which is a modified version of the SQ oracle.

Definition 10 (CSQ Oracle)**.** *For any query function $\Psi : \mathcal{X} \mapsto \{\pm 1\}$, and tolerance τ, let $P_\Psi = E[\Psi(x)\cdot c(x)]$. The CSQ Oracle returns $\hat{P}_\Psi \in [P_{\Psi-\tau}, P_{\Psi+\tau}]$*

In the proof of Theorem 13 (SQ learnability implies learnability with RCN) in Lecture 6, we showed that a SQ Oracle can be simulated using target-independent queries and CSQ queries. Since the distribution D is fixed to be the uniform distribution, any target-independence query (which only depends on x) can be simulated by sampling unlabeled datapoints from $D = U$. Therefore, if there is an algorithm which efficiently learns parities with an SQ oracle, then there is an algorithm which efficiently learns parities with a CSQ oracle. Therefore we will show that it is not possible to efficiently learn parities with a CSQ oracle.

The proof will use some techniques from analysis of Boolean functions. Analysis of Boolean functions is quite useful to in learning theory, particularly for Boolean valued functions such as parity.

To begin, we first think of any function f: $\{\pm 1\}^d \mapsto \{\pm 1\}$ as a vector $\vec{f}$ with 2^d entries:

$$\left(\frac{1}{2^{d/2}}f(-1,-1,\ldots,-1), \frac{1}{2^{d/2}}f(-1,-1,\ldots,1), \ldots \frac{1}{2^{d/2}}f(1,1,\ldots,1).\right)$$

Note that $\langle \vec{f}, \vec{g}\rangle = \mathop{\mathbb{E}}_{x\sim U}[f(x)\cdot g(x)]$ because,

$$\begin{aligned}\langle \vec{f}, \vec{g}\rangle &= \sum_{x\in\{\pm 1\}^d} \frac{1}{2^{d/2}}f(x)\cdot\frac{1}{2^{d/2}}g(x)\\ &= \mathop{\mathbb{E}}_{x\sim U}[f(x)\cdot g(x)].\end{aligned}$$

We also note that $\langle \vec{f}, \vec{f}\rangle = 1$.

A key technique used in understanding Boolean functions is to change the basis in which we investigate the properties of the function. Instead of writing the function in the original coordinate system, we will write it in the *Fourier basis*, which is given by the parity functions. Looking at the function in the Fourier basis can reveal many useful properties of the function. Here, our goal is going to be rather limited, and we will show that going to the Fourier basis implies that SQ queries about the parity function are likely not informative.

Recall that an orthonormal basis for a vector space is a set of orthogonal unit vectors that span the space. If $v_1,\ v_2$ are an orthonormal basis for $\mathbb{R}^2$, we can write any vector w as

$$w = \langle w, v_1\rangle v_1 + \langle w, v_2, v\rangle_2.$$

Claim 11. *The set of all parity functions form an orthogonal basis for* $\mathbb{R}^{2^d}$.

Proof. Note that for $S \neq T$.

$$\langle \overrightarrow{c_S}(x), \overrightarrow{c_T}(x)\rangle = \mathbb{E}[c_S(x) \cdot c_T(x)] = 0.$$

Also,

$$\langle c_S(x), \overrightarrow{c_S}(x)\rangle = 1, \forall s.$$

Therefore since the set of parity functions are orthogonal, unit norm, and comprise of 2^d functions, they are an orthonormal basis for $\mathbb{R}^{2^d}$. ■

Now for any CSQ $\Psi : \{\pm 1\}^d \mapsto \{\pm 1\}$,

$$\vec{\Psi} = \sum_{S: S\subseteq 1\ldots d} \alpha_S \overrightarrow{c_S}$$

where $\overrightarrow{c_S}$ is the parity function over S. Note that $\alpha_S = \langle \vec{\Psi}, \overrightarrow{c_S}\rangle = \mathop{\mathbb{E}}_{x\sim U}[\Psi(x) \cdot c_S(x)]$. Therefore if the target function is $c_S(x)$, the expected response P_Ψ to the CSQ Ψ is $P_\Psi = \alpha_S$.

Since $\langle \vec{\Psi}, \vec{\Psi}\rangle = 1$, $\sum \alpha_S^2 = 1$. Therefore (by Markov's inequality), there can be almost $\frac{1}{\tau^2}$ subsets S s.t. $|\alpha_S| \geq \tau$.

Note that if target is S^*, then CSQ oracle can just answer 0 to the query Ψ if $|\alpha_S^*| < \tau$. Since we draw the target parity function (the set S^* uniformly at random from all possible subsets, we have the following claim.

Lemma 12. *If the algorithm makes less than* $\Omega(\tau^2 2^d)$ *queries, the CSQ oracle can answer 0 to all these queries with probability at least* 0.99 *over choice of* S^*.

Proof. As we argued before, for any query Ψ there are at most $1/\tau^2$ subsets S such that $|\alpha_S| \geq \tau$. The oracle will respond with 0 if the query is such that $|\alpha_{S^*}| < \tau$. Therefore with each query Ψ, the algorithm can "check" if S^* belongs to some collection of at most $1/\tau^2$ subsets S. The algorithm gets 0 as a response if S^* is not one of the subsets in this collection. For simplicity, assume for now that the learning algorithm is deterministic. Then the algorithm checks groups of at most $1/\tau^2$ subsets with every query, and makes a fixed sequence of queries till it gets a non-zero response. If the algorithm makes less than $\tau^2 2^d/100$ queries, then it checks at most $2^d/100$ subsets, and since S^* is chosen uniformly at random the probability of it being one of these $2^d/100$ subsets is $1/100$. Therefore with probability at least 0.99, the algorithm gets a 0 as a response to each query. A similar argument can be made for randomized learning algorithms as well. ■

To complete the proof, we need to show that if the algorithm cannot succeed if it only receives 0 as a response. You can prove this by arguing that there are many such subsets S which are consistent with only receiving a 0 as a response to every query, and the algorithm has not received any information which distinguishes between different subsets S which are consistent with a 0 response. Since parity functions on any two different subsets are uncorrelated, if there are two different parity functions which are both perfectly consistent with all the responses, the algorithm cannot succeed since it cannot get low error on both of them.

■

Therefore, we cannot hope to efficiently learn parities in the SQ model. This implies that the framework we showed last time for showing learnability with RCN via SQ learnability cannot work for parities. What can we say then about learning parity with RCN?

Note that information theoretically (in terms of the required number of samples), learning parities with noise is still easy.

Theorem 13. *Parities are learnable (not necessarily efficiently) over the uniform distribution in the presence of RCN with noise level η with $O\left(\frac{d}{(1-2n)^2}\right)$ samples, with high probability.*

Proof. Let $\epsilon = \frac{1}{2} - \eta$. Note that the label for any example is preserved with probability $\frac{1}{2} + \epsilon$, and is flipped with probability $\frac{1}{2} - \epsilon$. Suppose we get $m = O(d/\epsilon^2)$ samples from the noisy example oracle $EX^{\eta}(c, D)$. Then using Chernoff/Hoeffding bounds and the property that parity functions are uncorrelated you can show that with high probability,

- The target function c_{S^*} is consistent with $> \left(\frac{1}{2} + \frac{\epsilon}{2}\right)$ fraction of examples.
- Any c_S with $S \neq S^*$ is consistent with $< \left(\frac{1}{2} + \frac{\epsilon}{2}\right)$ fraction of examples.

This shows that with high probability ERM will find the target function c_{S^*}. ■

The previous argument can be generalized to any class which can be PAC learned in the absence of noise, such as a concept class with finite VC dimension. Therefore, information theoretically, random classification noise does not change learnability.

However, it appears that the picture is quite different when we consider computational efficiency. We have shown that SQ algorithms cannot learn efficiently parities with noise, and it is widely suspected that there are no efficient algorithms at all for learning parity with noise. The best known algorithm for learning parity with noise runs in $2^{\frac{d}{\log d}}$ time, due to a result by [1]. Therefore the algorithm is just very slightly sub-exponential, and no further improvements are known. Learning parity with noise (LPN) is believed to be hard, and in fact a generalization of LPN known as Learning with Errors (LWE) forms the basis of cryptographic systems [2].

The $2^{\frac{d}{\log d}}$ time algorithm of [1] is not a SQ algorithm (and it cannot be, because of the SQ lower bound we showed for parities). Even though it is only slightly sub-exponential, this algorithm does imply that it is possible to learn a sub-class of parity functions in polynomial time. Consider the class of parity functions, but where the subset of coordinates S in the parity must belong to the first $O(\log d \log\log d)$ coordinates of the input (in other words, the parity must be supported on the first $O(\log d \log\log d)$ coordinates). You can show that this class also does not have polynomial SQ dimension, and hence cannot be learned efficiently in the SQ model. However, it can still be learned in polynomial time, due to the algorithm of [1]. This implies that there is a concept class which can be efficiently learned in the presences of noise, but is not efficiently learnable in the SQ model. Therefore SQ learnability is only sufficient, but not necessary, for PAC learnability with RCN.

More formally, our current understanding is the following,

$$\text{SQ-learnable concepts } \mathcal{C} \subset \text{PAC with RCN-learnable concepts } \mathcal{C} \subseteq \text{PAC-learnable concepts } \mathcal{C}.$$

Conditioned on hardness of LPN, the final inclusion is proper.

4 Computational-Statistical Tradeoffs

In the first part of the class, we studied the statistical complexity of learning, i.e. how much data do we need in order to learn? After that, we switched to computational aspects, and asked when learning is possible in polynomial time. A natural, and very active line of research, is to understand if and when there are tradeoffs between computational and statistical resources. When does computational efficiency come at the cost of statistical power?

One of the most studied problem in this direction is the planted clique problem.

Definition 14 (Planted Clique Problem)**.** *Given a graph generated from one of the following 2 distributions, decide which distribution generated the graph*

1. *$G(n, 1/2)$ denoting Erdös-Renyi graph with n vertices and probability of $\frac{1}{2}$.*
2. *Generate an instance of $G(n, 1/2)$ and plant a clique on k randomly chosen vertices of the graph.*

The notion of statistical power of an algorithm here is the minimum clique which can be detected by the algorithm.

Question: What is the smallest k at which these two distributions are information-theoretically distinguishable?

The following Lemma helps us understand the information theoretic limits here, i.e. the best statistical power among all possible algorithms.

Lemma 15. *An Erdös-Renyi random graph $G(n, 1/2)$ does not have a clique of $k \geq 3\log n$ w.h.p.*

Proof. Consider any subset of $k = 3\log n$ vertices.

$$\Pr(\exists \text{ a clique on these } k \text{ vertices}) = \left(\frac{1}{2}\right)^{\binom{k}{2}}. \tag{1}$$

By a union bound,

$$\Pr(\exists \text{ a clique on any subset of } k \text{ vertices}) \leq \binom{n}{k}\left(\frac{1}{2}\right)^{\frac{k \cdot k-1}{2}}. \tag{2}$$

Using the relation that $\binom{n}{k} \leq (\frac{ne}{k})^k$ and $k = 3\log n$,

$$\begin{aligned}\Pr(\exists \text{ a clique on any subset of } k \text{ vertices}) &\leq \left(\frac{ne}{k}\right)^k \left(\frac{1}{2}\right)^{\frac{k \cdot k-1}{2}} \\ &\leq \left(\frac{ne}{3\log n}\right)^{3\log n} \left(\frac{1}{n^3}\right)^{\frac{3\log n-1}{2}} \\ &= \left(\frac{ne}{3\log n}\right)^{3\log n} \left(\frac{1}{n^{1.5}}\right)^{3\log n-1} \\ &\leq \frac{1}{n}.\end{aligned}$$

■

Using this Lemma, we have an inefficient brute force search algorithm for $k \geq 3\log n$:

Algorithm 1 Brute-Force Search

```
  Search every subset of $k$ vertices
  **if** $\exists$ a clique on any subset **then**
      return (graph comes from a planted clique mode)
  **else**
      return (graph comes from $G(n, 1/2)$)
  **end if**
```

The running time of the algorithm is $k^2\binom{n}{k} = \Omega(n^{\log n})$, hence it does not run in polynomial time. But it does prove that it is information-theoretically possible to detect the planted clique for $k \geq 3\log n$.

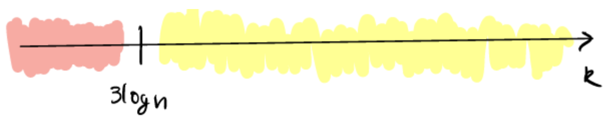

Figure 1: For $k << \log n$ it is information-theoretically impossible to distinguish the two cases but if $k \geq 3\log n$, information-theoretically it is possible to distinguish.

Question: When can we do this efficiently?

There is in fact a simple, efficient algorithm when $k \gg \sqrt{n\log n}$, based on the following simple Lemma.

Lemma 16. *The number of edges in an Erdös-Renyi random graph $G(n, 1/2)$ lies in* $\left[\frac{n(n-1)}{4} - 100\sqrt{\log n}\ ,\ \frac{n(n-1)}{4} + 100\sqrt{\log n}\right]$ *w.h.p.*

By adding a clique on k vertices we will add $\approx \binom{k}{2}\frac{1}{2} = \frac{k(k-1)}{4}$ edges (repeat previous lemma to do this rigorously).

$k = \sqrt{cn\log n}$ for some large c, we will add $\approx \frac{cn\log n}{4}$ edges.

- W.h.p., for $k = \sqrt{cn\log n}$, $G(n, 1/2)$ graph with planted clique has at least $\frac{n(n-1)}{4} + \frac{1000n\log n}{4}$ edges for large enough c (say $c = 1000$).
- For $k \gg \sqrt{n\log n}$ we can efficiently detect the clique w.h.p. by counting the number of edges in the graph.
- For $k \gg \sqrt{n}$ there is an efficient algorithm based on eigenvalue decomposition of the adjacency matrix of the graph.

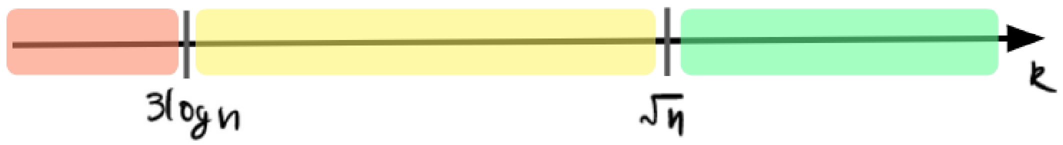

Figure 2: It is believed to be impossible to efficiently detect the presence of a planted clique in the yellow region ($3\log n \leq k \leq \sqrt{n}$). For the green region ($k >> \sqrt{n}$) efficient algorithms do exist. The planted clique conjecture makes this precise.

Definition 17 (Planted Clique Conjecture)**.** *There is no efficient algorithm to detect a planted clique of size $k = o(\sqrt{n})$ in a $G(n, 1/2)$ graph.*

This conjecture is still open, but is known to be true for restricted class of algorithms such as

- A version of SQ algorithms [3].
- Generalization of Semi-Definite Programs (SDPs) [4].
- Markov Chain Monte-Carlo (MCMC) methods [5].
- ... and many other algorithmic frameworks.

5 Further Reading

These lecture slides by Avrim Blum [lecture 1], [lecture 2] are good for additional reading about SQ dimension and SQ learnability. You can read this paper [6] for a more recent treatment of the topic. The survey paper [7] is a good reference for further reading about the SQ model, SQ dimension and SQ learnability.

Computational statistical tradeoffs are an active area of research, and this Simons program has talks on recent developments. You can read more about the planted clique problem, and also find the spectral algorithm which works at $k = \sqrt{n}$ and its analysis here.

References

[1] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.

[2] Oded Regev. The learning with errors problem. *Invited survey in CCC*, 7(30):11, 2010.

[3] Vitaly Feldman, Elena Grigorescu, Lev Reyzin, Santosh S Vempala, and Ying Xiao. Statistical algorithms and a lower bound for detecting planted cliques. *Journal of the ACM (JACM)*, 64(2):1–37, 2017.

[4] Boaz Barak, Samuel Hopkins, Jonathan Kelner, Pravesh K Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. *SIAM Journal on Computing*, 48(2):687–735, 2019.

[5] Mark Jerrum. Large cliques elude the metropolis process. *Random Structures & Algorithms*, 3(4):347–359, 1992.

[6] Vitaly Feldman. A complete characterization of statistical query learning with applications to evolvability. *Journal of Computer and System Sciences*, 78(5):1444–1459, 2012.

[7] Lev Reyzin. Statistical queries and statistical algorithms: Foundations and applications. *arXiv preprint arXiv:2004.00557*, 2020.