CSE 291 Scribe Notes Lecture 27

Mingshan Wang

June 5, 2017

1 Introduction

Recently we have been talking about robust statistics and high dimension algorithms. We will shift our focus to the lower bound of those algorithms. Most of the lower bounds covered in this class is information theoretic. However, those information theoretic lower bounds are a little bit better than what we can do computationally. In order to reduce the complexity of computing those lower bounds, we want to restrict our computation model with the hope to prove information theoretic lower bound more easily. In order to achieve that, we need to restrict the way the algorithm access the data, if the algorithm can't see all of the data points, the algorithm can do just fine because of the old information theoretic optimum. We need some reasonable restrictions on how our algorithm actually use the data. The algorithms we have been using involves computing moments, or compute moments after throw away some points that obey some conditions. But basically what they are doing is that they are taking averages of some functions of the data. But we can't quiet allow any function of the data because the function could be really big at the some values, or the values of the function could exactly encode the points up to some precision. We need some sorts of normalization.

2 Today: Statistical Query Algorithms

2.1 Introduction

The idea is that if we have function f : $Domain \rightarrow [0, 1]$, then we can approximate the expectation of f E[f(x)] on our data. In particular, we define a STAT(ϵ) oracle that given f : $Domain \rightarrow [0, 1]$, the oracle gives back the value $\mathrm{E}[f(x)] \pm \epsilon$. You are given some number that is guarantee to be in this range, and no other guarantees. You can simulate N calls to STAT(ϵ) using logN/ ϵ^2 samples and N time. Essentially, we are going to use this methodology to prove lower bounds in this statistical query model. Those lower bounds requires we either need to take statistical query with very small ϵ, thus we will need a lot of samples to get those or need to use many queries, which corresponds the algorithm to have very long runtime. So those lower bounds don't work for general algorithm, but this model is reasonably good. The model has many good properties, for example, these statistical algorithms are very robust against noise. In particular, you can deal with $\epsilon/2$ adversarial noise, thus the adversarial noise won't affect the ability to get those statistically query accuracy of ϵ. It can also deal with unlimited random noise. For example, with 90% of probability the sample is replaced by a sample from a known distribution Y. $E[0.9Y + 0.1X] = 0.9E[f(Y)] + 0.1E[f(X)]$. And if you know what Y is, you can delete all the terms related to Y, and get what E[f(X)] is.

All the robust statistic filter algorithms we have been talking about, we can implement them in statistical query model. We can do all the testing algorithms under the condition that within bounded domain,since every query to the oracle, we are only given a finite number of elements.

2.2 Statistical Query Lower Bounds

We have a distribution x on F_2^n, that is the uniform distribution over a random (n-1) dimension subspace. The goal is to learn x. The statistical algorithm can do very well in this case. The idea is Fourier analysis. We want to know E[f(X)], and

$$f(x) = \sum_{s \in [n]} \hat{f}(x) X_s$$

and

$$X_s = \Pi_{i \in s}(-1)^{x_i}$$

If you look at the sum of the Fourier coefficients, they are bounded by 1.

$$\sum |\hat{f}(s)|^2 = \text{Average}(|f(x)^2|) \leq 1$$

The second thing to note is that if X = $\{X_s = 1\}_{unif}$, then E[f(x)] = $\hat{f}(\emptyset)$ + $\hat{f}$(s). The problem here is that almost all $|\hat{f}(s)|$ are small, so if we do the adversarial thing with our oracle, then we can have a oracle return $\hat{f}(\emptyset)$ with high probability. The point is if we have a random sample subspace, then whatever the query our algorithm is going to make, with high probability, $\hat{f}(s)$ is small, therefore with high probability our algorithm is just going to return $\hat{f}(\emptyset)$. If we keep returning $\hat{f}(\emptyset)$, our algorithm is never learned anything.

In particular, if STAT(ϵ) query, is going to be the case that at most $1/\epsilon^2$, $|\hat{f}(s)| > \epsilon$. That means oracle returns $\hat{f}(\emptyset)$ for all but $1/\epsilon^2$ possible Xs. That means if a random hyperspace, then we can know the result every query will make without calling our oracle. So it requires you make $\Omega(2^n \epsilon^2)$ queries to STAT(ϵ).

2.3 Generalization

The above example is far too specific, we want to generalize.The setting is that we had many distributions that we want to distinguish. The point is that those distributions were nearly orthogonal. We first need to define inner product.

Innerproduct: D is a distribution on the same domain.

$$< p, q >_D = \sum_x \frac{p(x)q(x)}{D} = \int \frac{dp * dq}{dD}$$

$< p, q >_D \geq 1$ because $< p, D >_D$=1 and $< q, D >_D$=1.

We need to subtract the uniform distribution off to allow distributions to become orthogonal. So what we actually want is

$$X_D(p, q) = < p - D, q - D >_D = < p, q >_D - 1 = \int \frac{dp * dq}{dD} - 1$$

and this is the inner product we will be using.

2.4 General Statistical Problem: Learning

Suppose we have a distribution P in some class C(finite set), we are trying to learn P exactly. Suppose there is another distribution D, $p_1, p_2, ...p_N \in$ C. Firstly we want $|X_D(p_i, p_j)| \leq \delta$ for i $\neq$ j, and secondly we also need that $X_D(p_i, p_j) < C$ for all i.
We have query f. We want for most of p$_i$, E[$f(p_i)$]≈ E[f(D)] where f(p_i) $=< p, f \cdot D >_D$ and f(D) = $< D, f \cdot D >_D$. We want $< p_i - D, f \cdot D >$ to be small.

Notice that f·D is a vector that can't be too big because $< f \cdot D, f \cdot D >= \int \frac{f^2 dD^2}{dD} = \int f^2 dD \leq 1$.

Suppose $| < f_D, p_i - D >_D | > \epsilon$ for i = 1,...m, consider g = f $\cdot D - x(\sum \pm(p_i - D))$.
And $< g, g >_D \geq 0$, also $< g, g >_D = < f \cdot D, f \cdot D >_D - x \sum_i^m < f \cdot D, \pm(p_i - D) >_D + x^2 (\sum_i^m < p_i - D, p_i - D >_D + \sum_{i \neq j} \pm < p_i - D, p_j - D >) \leq 1 - 2xm\epsilon + x^2(mc + m^2\delta)$, which is positive for all x.

This implies that $m\epsilon < \sqrt{mc + m^2\delta}$ and $\epsilon < \sqrt{c/m + \epsilon}$. In particular, if $m > c/\delta, \epsilon < \sqrt{2\delta}$.

3 Conclusion

$$\text{E}[f(p_i)] - \text{E}[f(D)] > \sqrt{2\delta}$$

for at most c/δ i's.

Remember our assumption was that $p_1, ..p_N, X_D(p_i, p_i) \leq C, X_D(p_j, p_j) \leq \delta$. Learning p_i requires N·δ/C queries to STAT($\sqrt{2\delta}$) oracle. If N is large, that indicates we need substantial number of queries. Above is the basic for statistical query lower bound.

3.1 Only obstacle

Suppose finite class C = $p_1, ...p_N$, SQ-dimension max d such that there is a distribution D, and $q_1, ..q_d \in$ C such that $X_D(q_i, q_i) \leq$d and $X_D(q_i, q_j) \leq 1/$d.If SQ-dim is big and N is large, then there aren't any good statistical query algorithms. However of SQ-dim = d, there exists a statistical query algorithm uses poly(d)logN queries to STAT(poly(1/d)) oracles.