CS6781 - Theoretical Foundations of Machine Learning

Lecture 23: Learning in the Presence of Noise

April 30, 2020

Lecturer: Nika Haghtalab *Readings: N/A*
Scribe: Ritabrata Ray

In the previous lectures, we encountered several models of learning:

- Realizable PAC learning: In this model, all data is labelled by some $h^* \in \mathcal{H}$ and we find $h \in \mathcal{H}$ such that $\text{err}_{\mathcal{D}}(h) \leq \epsilon$.
- Agnostic PAC learning: Here no assumption is made on how data is labelled and we aim to find an $h \in \mathcal{H}$ such that $\text{err}_{\mathcal{D}}(h) \leq \min_{h^* \in \mathcal{H}} \text{err}_{\mathcal{D}}(h^*) + \epsilon$.

In this lecture, we will explore another model of learning in the *presence of noise*. To motivate this model we can think of the agnostic PAC model as if the data was labelled according to $h^* \in \mathcal{H}$, an adversary flips the labels of some OPT fraction of the data and we try to match the predictions of h^*. This flipping of the labels can be interpreted as *noise*. In this lecture, we consider a different model of noise that is more benign, where the label of every instance is flipped with equal probability.

1 Random Classification Noise (RCN) Model

In this lecture we seek a middle ground between realizable PAC and the adversarial noise models, called the RCN model.

Definition 1.1. A **random classification noise (RCN)** oracle of rate $\eta < \frac{1}{2}$ for a fixed hypothesis $h^* : \mathcal{X} \to \{-1, +1\}$ and distribution $\mathcal{D}$ over domain $\mathcal{X}$ (the marginal distribution of $\mathcal{X}$ as in the realizable PAC model) is denoted by $\text{EX}^{\eta}(h^*, \mathcal{D})$. When we call $\text{EX}^{\eta}(h^*, \mathcal{D})$ it draws $x \sim \mathcal{D}$ and with probability $1 - \eta$ gives us $(x, h^*(x))$ and with probability η it gives us $(x, -h^*(x))$.

Some remarks on the RCN model:

- Here the adversary does not choose which points are flipped but rather each instance has the same likelihood of getting its label flipped.
- If $\eta = 0$ then this is just taking $iid\ x_1, \ldots, x_m \sim \mathcal{D}$ and labelling them by h^* which is the realizable PAC model.
- The noise is independent across all calls to the oracle $\text{EX}^{\eta}(h^*, \mathcal{D})$ and the noise rate is exactly η for all points rather than the more optimistic model of at most η.

- If $\eta = \frac{1}{2}$, then we have pure noise and both h^* and $-h^*$ and in fact any $h \in \mathcal{H}$ is equally good and there is nothing to learn. As $\eta \to \frac{1}{2}$ detecting between h^* and $-h^*$ becomes harder both with respect to the runtime and the sample complexity as they grow with $\frac{1}{1-2\eta}$. (This is similar to the situation encountered earlier in the course while proving lower bounds on the sample complexity of Agnostic PAC learning).

Next we define the concept of learnability in this model.

Definition 1.2. PAC learning in the presence of RCN: A hypothesis class $\mathcal{H}$ is PAC learnable in the presence of RCN if there is an algorithm $\mathcal{A}$ such that for any $h^* \in \mathcal{H}$, any distribution $\mathcal{D}$, and any $\epsilon > 0$, $\delta > 0$ and any (known) $\eta < \frac{1}{2}$, given access to $\text{EX}^\eta(h^*, \mathcal{D})$ with probability $> 1 - \delta$ gives h such that $\text{err}_{\mathcal{D}}(h) \leq \epsilon$. Moreover we say $\mathcal{H}$ is efficiently learnable if algorithm $\mathcal{A}$ runs in time $poly(\frac{1}{\epsilon}, \frac{1}{\delta}, \frac{1}{1-2\eta}$, size of natural representation of $\mathcal{H})$.

Remark 1.3. The assumption that η is known is not required but we make it for simplicity in the forthcoming analysis. Also by $\text{err}_{\mathcal{D}}(h)$ we mean the error of h when $x \sim \mathcal{D}$ and h^* then labels the instances x. We call this distribution $\mathcal{D}$ the *clean* distribution.

In this model, we are measuring success with respect to the clean distribution $\mathcal{D}$ where we never actually see samples coming from this distribution. So, the question is how can we learn in such a strong model?
We introduce a new notation $\mathcal{D}(\eta)$ as the noisy distribution of the data where we see data from the RCN oracle $\text{EX}^\eta(h^*, \mathcal{D})$ and $\mathcal{D}$ is the clean distribution. We begin with the observation that if we can estimate $\text{err}_{\mathcal{D}(\eta)}(h)$ then we can also measure $\text{err}_{\mathcal{D}}(h)$ with the same accuracy since

$$\text{err}_{\mathcal{D}(\eta)} = \text{err}_{\mathcal{D}}(h)(1-\eta) + (1 - \text{err}_{\mathcal{D}}(h))\eta.$$

The above equation is true since the hypothesis h mis-classifies an instance coming from the noisy distribution if either the true label was received but the hypothesis would have mis-classified it even if it had arrived from the clean distribution or if the hypothesis labels the instance correctly coming from the clean distribution but the noise had the label flipped so it accounts for a mis-classification in the noisy distribution. Note that the above equation is only true if each label has the same probability of flipping η (instead of any label being flipped with probability at most η), as is the case with the RCN model.

2 Learning Monotone Conjunctions in the Presence of RCN

To answer the question of learning efficiently in this model we will consider the problem of learning *monotone conjunctions*. We had visited the problem of learning monotone conjunctions in the consistency model (realizable PAC setting) during the early parts of this course. Let us recall that in this problem we have $\mathcal{X} = \{0,1\}^n$, $\mathcal{Y} = \{0,1\}$ and $\mathcal{H} = \{h : h(\mathbf{x}) = \bigwedge_{i \in J} x_i, \;\; J \subseteq [n]\}$ and in the consistency model all the samples are labelled according to a fixed $h^* \in \mathcal{H}$. Recall that the algorithm for learning monotone conjunctions in the consistency model was to begin with

$h(\mathbf{x}) = \bigwedge_{i\in[n]} x_i$ and go one by one on $S = \{\mathbf{x_1}, \ldots, \mathbf{x_m}\}$ and if we see $x_i = 0$ but $h^*(\mathbf{x}) = 1$ for some $\mathbf{x} \in S$, then we eliminate i from J. However this algorithm will not work in the RCN model as we may kick out some x_i which was actually in h^* because we received a flipped label. Most likely, this algorithm will kick out all $i \in J$ where J corresponds to the true hypothesis h^*.
We use another algorithm to learn monotone conjunctions in the RCN model. At a high level, we focus less on individuals and focus on population statistics. We consider the bad condition: $x_i = 0$, $h^*(\mathbf{x}) = 1$ and ask what is the probability with which this bad condition occurs! Informally, we output $h(\mathbf{x}) = \bigwedge_{i\in J} x_i$ where J contains all such $i \in [n]$ which satisfy: $\Pr[x_i = 0, h^*(\mathbf{x}) = 1] = 0$ (or close to zero) but have large $\Pr[x_i = 0]$, so that the probability of keeping bad i's in our J is low.

Let us first describe this algorithm as an alternative the elimination algorithm above even when the distribution is not noisy. We begin with the following definitions:

Definition 2.1. **Significant variable:** A variable x_i is called a significant variable if

$$\Pr_{x\sim\mathcal{D}}[x_i = 0] \geq \frac{\epsilon}{4n}$$

where n is the number of boolean variables.

Definition 2.2. **Harmful variable:** a variable x_i is called harmful if

$$\Pr_{x\sim\mathcal{D}}[x_i = 0,\ h^*(\mathbf{x}) = 1] > \frac{\epsilon}{4n}.$$

The algorithm then is predicting using: $h(\mathbf{x}) = \bigwedge_{i\in J} x_i$ where $J = \{i : i \text{ is significant but not harmful}\}$.
We have for this h,

$$\mathrm{err}_{\mathcal{D}}(h) = \Pr_{x\sim\mathcal{D}}[h(\mathbf{x}) = 0, h^*(\mathbf{x}) = 1] + \Pr_{x\sim\mathcal{D}}[h(\mathbf{x}) = 1, h^*(\mathbf{x}) = 0]$$

In the above equation, the first error term occurs due to the presence of a variable x_i in h such that x_i is not in h^*. As x_i is in h by definition , it is a significant but not a harmful variable. Any variable x_i which is not harmful can cause the event $x_i = 0$, $h^*(\mathbf{x}) = 1$ with probability $\leq \frac{\epsilon}{4n}$. Since any of the n variables can be such a significant but not harmful variable, by union bound $\Pr_{x\sim\mathcal{D}}[h(\mathbf{x}) = 0, h^*(\mathbf{x}) = 1] \leq n \times \frac{\epsilon}{4n} = \frac{\epsilon}{4}$.
The second error term in the above equation is caused by variable x_i which is present in h^* but was not added in h. Since it is present in h^*, it cannot be *harmful*, but as it is not in h it is not *significant*. The error occurs when this *insignificant* variable x_i is 0, but insignificant variables are assigned 0 in $\mathcal{D}$ with probability at most $\frac{\epsilon}{4n}$. Again as any of the n variables can be such an *insignificant* variable, by the union bound, $\Pr_{x\sim\mathcal{D}}[h(\mathbf{x}) = 1, h^*(\mathbf{x}) = 0] < n \times \frac{\epsilon}{4n} = \frac{\epsilon}{4}$. Thus,

$$\mathrm{err}_{\mathcal{D}}(h) = \Pr_{x\sim\mathcal{D}}[h(\mathbf{x}) = 0, h^*(\mathbf{x}) = 1] + \Pr_{x\sim\mathcal{D}}[h(\mathbf{x}) = 1, h^*(\mathbf{x}) = 0] \leq \frac{\epsilon}{4} + \frac{\epsilon}{4} = \frac{\epsilon}{2}.$$

Remark 2.3. The reason we have $\mathrm{err}_{\mathcal{D}}(h) \leq \frac{\epsilon}{2}$ instead of ϵ is to keep room for estimation errors when we use empirical estimates to set the thresholds for significant and harmful variables.

Now we consider the problem of learning monotone conjunctions in the presence of RCN. We use the above algorithm for the noiseless case but this time we don't have the true values of $P(x_i = 0) = \Pr_{x\sim\mathcal{D}}[x_i = 0]$ and $P^1(x_i = 0) = \Pr_{x\sim\mathcal{D}}[x_i = 0,\ h^*(\mathbf{x}) = 1]$ to determine which variables are significant and which are harmful. So, we estimate both these quantities.

To estimate $P(x_i = 0)$, we take a number of samples from the oracle $\mathrm{EX}^{\eta}(h^*, \mathcal{D})$ and throw out the labels. Then we do the empirical average of the cases where $x_i = 0$ to estimate $P(x_i = 0)$ and by Chernoff bound for a sufficiently large number of samples $\tilde{O}(\frac{n}{\epsilon})$, this empirical average is within a factor 2 of the true value of $P(x_i = 0)$.

In order to estimate $P^1(x_i = 0)$, we use

$$P^1(x_i = 0) = \Pr[x_i = 0].\Pr[h^*(\mathbf{x}) = 1|x_i = 0]$$

Here for the first term in the RHS we use the above estimated $P(x_i = 0)$ and for the second term first of all we take some number of samples from $\mathrm{EX}^{\eta}(h^*, \mathcal{D})$ and take the empirical fraction of those labelled 1 among the ones having $x_i = 0$. By Hoeffding bound, for a sufficiently large number of samples this empirical average is close to $\Pr_{(x,y)\sim\mathcal{D}(\eta)}[y = 1|x_i = 0]$. But in the RCN model we have, as seen earlier:

$$\Pr_{(x,y)\sim\mathcal{D}(\eta)}[y = 1|x_i = 0] = (1-\eta)\Pr_{x\sim\mathcal{D}}[h^*(\mathbf{x}) = 1|x_i = 0] + (1 - \Pr_{x\sim\mathcal{D}}[h^*(\mathbf{x}) = 1|x_i = 0])\eta$$

Now having an estimate for the LHS we can estimate $\Pr_{x\sim\mathcal{D}}[h^*(\mathbf{x}) = 1|x_i = 0]$ with the same accuracy multiplied by a factor of $\frac{1}{1-2\eta}$ by solving for it from the above equation. Now with these two estimates we have an estimate for $P^1(x_i = 0)$.
Thus, using this algorithm monotone conjunctions are PAC learnable in the RCN model.

3 Statistical Query Model

Observe that the algorithm for learning monotone conjunctions in the RCN model (last section) needed the estimates for $P(x_i = 0)$ and $P^1(x_i = 0)$ for all $i \in [n]$. If we have these estimates then we no longer need an access to the oracle but we can still PAC learn monotone conjunctions in the RCN model. In other words, this algorithm did not use information about individual points but rather used population statistics. This is useful when learning is done in presence of random noise, as noise present in one individual is a random bad event that should not have large impact on what the algorithm does. As we will see later, this is also important when learning is done while keeping privacy issues in mind as individual queries on the data points are more susceptible to a breach of privacy than queries such as population statistics. The model we develop here formalizes this use of population level statistics. This is called the **Statistical Query Model**. This model puts restrictions on how we can use $\mathrm{EX}^{\eta}(h^*, \mathcal{D})$ to only compute population level statistics.
Here we have another (a different) oracle:

Definition 3.1. $\mathbf{STAT}(\mathbf{h}^*, \mathcal{D})$ which accepts queries of the form (ψ, τ), where $\psi : \mathcal{X} \times \{0,1\} \to \{0,1\}$ (is a function that takes as input an unlabelled instance and a bit and outputs a bit) and $0 < \tau \leq 1$ (is the *tolerance* value) and returns a value v such that

$$|v - \mathbb{E}[\psi(x, h^*(x))]| \leq \tau$$

We can interpret $\psi(x, h^*(x))$ as a predicate or a property of $x \in \mathcal{X}$. We use it as whether the property is (un)satisfied by $(\mathbf{x}, h^*(\mathbf{x}))$ and τ is the tolerance or accuracy of the query. An example of such a query is $\psi_i(\mathbf{x}, h^*(\mathbf{x})) = \mathbf{1}\{h^*(\mathbf{x}) = 1 \text{ and } x_i = 0\}$

3.1 Example-Monotone Conjunctions in the Statistical Query Model

We take the the algorithm for learning monotone conjunctions in the RCN model from the previous section as an example. To estimate $\Pr_{\mathbf{x}\sim\mathcal{D}}[x_i = 0]$ we make the following query to $STAT(h^*, \mathcal{D})$:

$$\psi_i(\mathbf{x}, b) = \begin{cases} 1 \text{ if } x_i = 0 \\ 0 \text{ if } x_i = 1 \end{cases}$$

Here b is a dummy bit and has no effect on the outcome. We have $\mathbb{E}[\psi_i(\mathbf{x}, h^*(\mathbf{x}))] = \Pr_{\mathbf{x}\sim\mathcal{D}}[x_i = 0]$, and we get the desired estimate in expectation with a tolerance of τ.

Similarly in order to estimate $\Pr_{\mathbf{x}\sim\mathcal{D}}[x_i = 0, h^*(\mathbf{x}) = 1]$ we make the query $\overline{\psi_i}(\mathbf{x}, h^*(\mathbf{x})$ to $STAT(h^*, \mathcal{D})$ where:

$$\overline{\psi_i}(\mathbf{x}, b) = \begin{cases} 1 \text{ if } x_i = 0, b = 1 \\ 0 \text{ otherwise} \end{cases}$$

Again, $\mathbb{E}[\overline{\psi_i}(\mathbf{x}, h^*(\mathbf{x})] = \Pr_{x\sim\mathcal{D}}[x_i = 0 \wedge h^*(\mathbf{x}) = 1]$. so, we again get the desired estimate in expectation within a tolerance of τ.

Thus, the analysis from the last section shows that we can learn monotone conjunctions in the statistical query model.

3.2 SQ-Learnability

Next we formally define the notion of learnability in this statistical query model.

Definition 3.2. **Learnability in the Statistical Query Model**: A hypothesis class $\mathcal{H}$ is efficiently learnable in the statistical query model if there is an algorithm $\mathcal{A}$ such that for every $\mathcal{D}$, $h^* \in \mathcal{H}$, $0 < \epsilon < 1$, which has access only to $STAT(h^*, \mathcal{D})$ learns a hypothesis h, such that $\mathrm{err}_{\mathcal{D}}(h) \leq \epsilon$ using queries (ψ, τ) such that $\tau > \frac{1}{\mathrm{poly}(\frac{1}{\epsilon}, \text{ size of natural representation of } \mathcal{H})}$ and ψ is evaluated in time $\mathrm{poly}(\frac{1}{\epsilon}, \text{ size of natural representation of } \mathcal{H})$ and $\mathcal{A}$ halts in $\mathrm{poly}(\frac{1}{\epsilon}, \text{ size of natural representation of } \mathcal{H})$ time.

As an example, the algorithm for learning monotone conjunctions in the statistical query model, from the previous subsection, is also efficient as the size of natural representation of $\mathcal{H}$ here is n and the run time for each query is also linear in n and doesn't depend on ϵ, thus $\mathrm{poly}(n, \frac{1}{\epsilon})$. Also

the algorithm halts in a polynomial time as it only makes 2 queries for each $i \in [n]$. Moreover, the analysis still hold in this model if we choose a tolerance level $\tau \geq \frac{\epsilon}{4n} = \frac{1}{\text{poly}(\frac{1}{\epsilon}, n)}$, thus satisfying all the requirements for efficient learning in the statistical query model.

We notice that there is no notion of δ in the statistical query model and it is actually the tolerance level τ that accounts for δ in this model. In the next class, we will see how to simulate these queries with a desired level of tolerance τ from random samples and how the number of samples is related to the probability $1 - \delta$ of achieving the desired level of tolerance τ.