1 Lecture 9: 2017.03.27

Some remarks from last time.

Remark 1.1 (Out of sample generalization)**.** *In particular, if we know that $\gamma_t \geq \gamma$, we have that a simpler bound for the training error: $\frac{1}{m}\left|\{i : H(x_i) \neq y_i\}\right|$. Then if we choose the number of rounds of Adaboosting, $T \geq \frac{1}{2\gamma^2}\ln(m)$, we can guarantee that H is consistent with the set S. Further, if $VCD(\mathcal{H}) = d$, then the VCD of $H's$ class is $\leq dT$ (linearity result for VCD-dimension, check the textbook). This means that if we choose T so that $T \geq \frac{1}{2\gamma^2}\ln(dT)$, then by VCD-theory we have good generalizations out of the sample S, i.e. with respect to the true distribution.*

Remark 1.2. *Adaboost provides a sort of converse to the following result: "hypothesis compression" $\implies$ "learning". If we have an algorithm that has a bad relation with the ε of the PAC-learning model, i.e. it outputs big hypothesis with respect to the error, then we can choose a bigger ε and feed this "bad" algorithm to obtain a much better hypothesis (In fact, sub-linear in ε).*

Today

- Learning with queries
- Algo for DFA
- PAC "solar system"

1.1 Learning with queries

PAC + Membership Queries

In essence, it is PAC-learning with a membership query oracle. This gives the algorithm to create "synthetic" examples and ask for the true labels. This assumes that there is an underlying ground truth, which might not be true in practice.

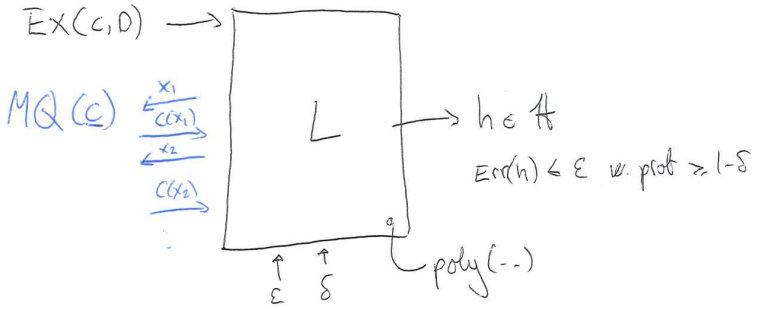

Figure 1: PAC with membership queries.

Remark 1.3. *On the surface, adding these membership queries are adding some power to the algorithm that cannot be simulated with $EX(c, D)$ or statistical queries.*

Exact Learning with Membership & Equivalence Queries

Now consider the following learning model where we have membership queries of individual inputs and an equivalence query oracle that answers whether an hypothesis is the same as the target class (as functions), c, and gives *some* counterexample if appropriate i.e. some input where they disagree.

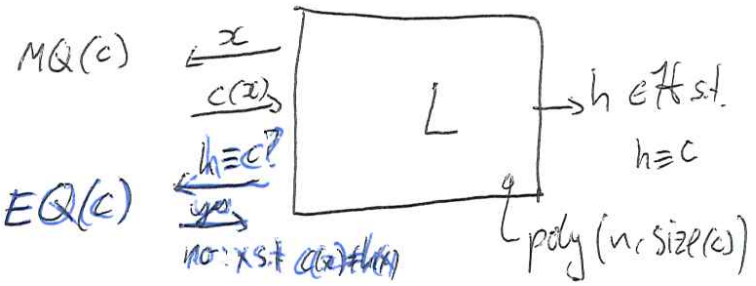

Figure 2: Exact learning with membership and equivalence queries.

Theorem 1.4. *If $\mathcal{C}$ is exactly learnable from MQ & EQ, then $\mathcal{C}$ is PAC+MQ-learnable.*

Sketch of proof. We can approximate EQ with enough samples from EX. In other words, we sample looking for a counterexample. If we find it great, we can return it. If we don't find it in a reasonable time, then also great because we already have a reasonably good hypothesis. □

1.1.1 Exact Learning of DFAs in MQ & EQ (Angluin)

First of all, recall that PAC-learning DFAs[1] is provably intractable assuming that factoring is hard.

Example 1.5 (DFA). Consider $X = \{0,1\}^*$ with labels $\{+,-\}$. i.e. arbitrary length strings of 1s and 0s. The goal is to learn in poly(size(c), longest counterexample).

[1] Deterministic Finite Automata.

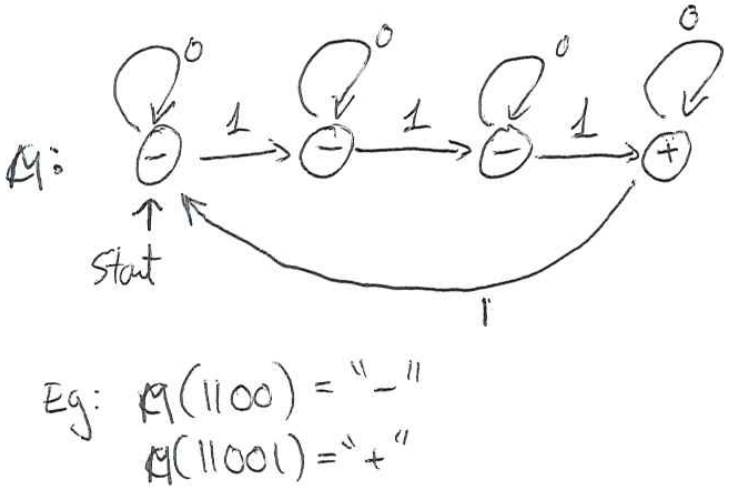

Figure 3: Example of DFA.

This function is actually $M(x) = +$ if and only if the #1s in x is $\equiv 3 \mod 4$.

Data structure: classification tree T

To be able to learn DFAs we use a binary tree satisfying the following properties.

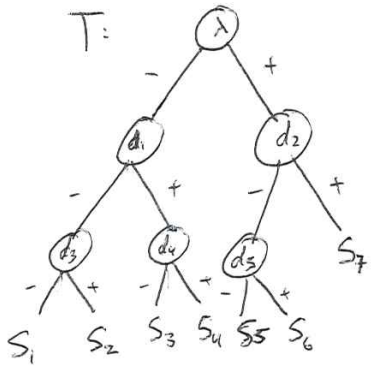

Figure 4: Classification tree.

- All the s and d are strings, λ is the empty string.
- Leaves s: state access strings.
- Internal nodes d: distinguishing strings.
- Invariants:
  - i) Each s_i reaches a different state of M.
  - ii) For all s_i and d_j, s_i is in left/right subtree of $d_j \iff s_i d_j$ reaches a $-/+$ state of M.

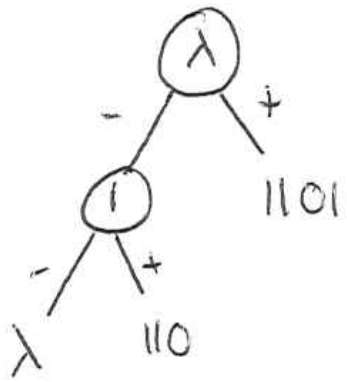

Figure 5: Classification tree for the example above.

Learning Model

1. MQ on λ to get the start state of λ i.e. what is the label of the start state, say $+$.

2. EQ on the trivial machine:

Figure 6: Trivial DFA.

If the machine is not trivial, we obtain a counterexample string, $\langle s, - \rangle$, and we can initialize the tree.

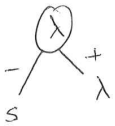

Figure 7: Initial classification tree.

3. Now we can iterate this process using equivalence queries and enlarging the tree. But before we can do that we need how to transform a classification tree into a hypothesis DFA and how to make progress on the tree given a new counterexample.

Definition 1.6 (Sift operation)**.** Given a classification tree, T, and a string of 1s and 0s, s, we define the sift of s by T, sift(T, s), as the leaf that is reached by the string s in the following way. At each node, we follow the string s and then the string of the node d_i in the true DFA machine and depending on the label of the state we reach, we go left/right down the tree. In the learning model we can actually do this using membership queries. For example:

- At the root, MQ the string $s\lambda$. Say that the query returns $+$, that means we go down the right node.
- MQ the string sd_2 and suppose the query returns $-$, we go down the left node.
- Repeat until we reach a leaf of the tree.

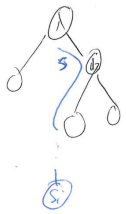

Figure 8: Sift operation.

Remark 1.7.

- *A tree T produces a partition of the space of all strings by labeling each string with* $\text{sift}(T, s)$.
- *Strings that reach the same state o the true machine must sift to the same leaf. This is because of the properties of the tree mentioned above.*

From a tree to a hypothesis machine $\widehat{M}$.

- For each leaf of the tree we have a state of the machine.
- To obtain the transitions between states we use the sift operation. For example, the state corresponding to the leaf s_i is connected to $\text{sift}(T, s_i0)$ and $\text{sift}(T, s_i1)$.

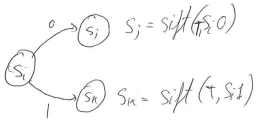

Figure 9: Transitions between states of $\widehat{M}$.

Making progress on the classification tree.
Once we have $\widehat{M}$ we can EQ and either be done or produce a counterexample. Suppose that the equivalence query on $\widehat{M}$ produces the counterexample $\gamma = \gamma_1\gamma_2 \dots \gamma_m$ where $\gamma_l \in \{0, 1\}$. For simplicity assume that γ is $+$ in the true machine M and $-$ in $\widehat{M}$. We use γ to improve the classification tree.

- As observed above, our current tree induces a partition of the space of strings, $\{0, 1\}^*$, and, equivalently, on the states of the machine M.
- Each of the partition pieces corresponds to one of the current leaves.

- We can assume that $S_1 = \lambda$, i.e. that the first leaf tracks the start state of the machine.
- Follow γ under $\widehat{M}$ and M and see where they diverge.

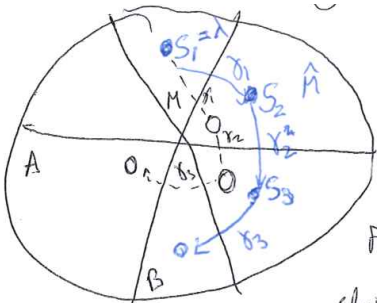

Figure 10: Split criterion for leaves.

Since they diverge, at some point we must end up in different equivalence classes which implies that on the step immediately before that we have two different states on the same partition piece. In the figure above, this happens after following $\gamma_1\gamma_2$. Thus, the string $\gamma_1\gamma_2$ reaches a *new state* that used to be in the same equivalence class as S_3. We can use this to split the leaf S_3.

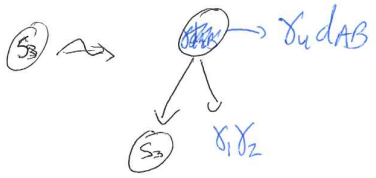

Figure 11: Splitting a leaf.

where d_{AB} is the distinguishing string of the least common ancestor of the leafs A and B, which are also leaves of our tree.

4. The stopping condition is when we have as many leaves as the true machine has states.

1.2 PAC-learning "Solar System"