CSE 513T Theory of Artificial Intelligence & Machine Learning January 27, 2015

Lecture 5

Lecturer: Brendan Juba *Scribe: Shilai Lu*

1 Recap

- "Proper Learning" labels given by $c \in C$ learn using a hypothesis $h \in C$.
- "Improper learning" learn using $h \in H \supset C$.
- 3-term DNF: OR of at most 3 ANDs of literals.
- Lemma: If there is a proper PAC-learning algorithm for a class of representation c, then there is a randomized poly-time algorithm for solving
$$Consis_c^{B(n,m)} = \{([(x^{(1)}, b^{(1)}), ..., (x^{(n)}, b^{(n)})], c) : c \in C, |c| \leq B(n,n), \forall i \;\; c(x^{(i)}) = b^{(i)}\}$$

2 Hardness of proper learning of 3-term DNF

In this section, we'll show $Consis^{3n}_{3-termDNF}$ is NP-complete. if "$P \neq NP$", then no NP-complete problem has a poly-time algorithm, which is an open question and is widlely believed. Regardless, we don't expect to come up with an algorithm.

2.1 To show a problem NP-complete

We'll show how to solve some other NP-complete problem using an algorithm for our problem.
Our NP-complete problem of the day: **The Graph 3-Coloring Problem**
Given as input an undirected graph $G = (V, E)$ with vertex set $V = \{1, ..., n\}$ and edge set $E \subseteq V \times V$, determine if there is an assignment of a color to each element of V such that at most 3 different colors are used, and for every edge $(i, j) \in E$, vertex i and vertex j are assigned different colors.

$$\text{3 Coloring} = \{(G, \mathcal{X}), \forall (u, v) \in E, \mathcal{X}(u) \neq \mathcal{X}(v)\}$$
$$\text{where } G = (V, E), \quad \mathcal{X}(\text{"}coloring''): V \to \{R, G, B\}$$

Pf:
Solving 3-Coloring using an algorithm for $Consis^{3n}_{3-termDNF}$. Construct an "interesting" data set that is labeled by a 3-term DNF to argue that the input graph is 3-colorable. Consider the following examples on $n = |V|$ attributes (number the vertices 1,...,n). For each $v(i) \in V$ with 0 in the ith position and 1's everywhere, we have one position example:

(111...101...1), 1←label

For each edge $(i, j) \in E$ with 0's in ith and jth positions, and 1's everywhere else, we have one negative example:

(11..101..101..1), 0←label

We'll show:

1. If the graph G is 3 colorable, there is a consistent 3-term DNF (since every 3-term DNF has size $\leq$ 3n, this is a solution to $Consis^{3n}_{3-termDNF}$).
2. If there is a solution to $Consis^{3n}_{3-termDNF}$, then we can extract a 3-coloring from the 3-term DNF produced by any algorithm.
3. Not necessarily the 3-term DNF we show exists in point 1.

Given a 3-coloring $\mathcal{X}$ of G, considering the 3-term DNF:

$$T_R = \wedge_{\forall i \mathcal{X}(i) \neq R}\ x_i,\ T_B = \wedge_{\forall i \mathcal{X}(i) \neq B}\ x_i,\ T_G = \wedge_{\forall i \mathcal{X}(i) \neq G}\ x_i \Longrightarrow T_R \vee T_B \vee T_G$$

Consider any of the positive examples, the example correspondent to the vertex i will be labeled by 1 since x_i doesn't appear in $T_{\mathcal{X}(i)}$ and the rest of the attributes are set to 1. So $T_{\mathcal{X}(i)} = 1$, and hence the DNF $T_R \vee T_B \vee T_G = 1$. On the other hand, consider any of the negative examples correspondent to an edge(i, j). Since the graph is assumed to be properly colored by $\mathcal{X}$ and $\mathcal{X}(i) \neq \mathcal{X}(j)$. We know that the terms not correspondent to $\mathcal{X}(i)$ contain x_i, and hence evaluate to 0 in these examples (since $x_i = 0$). Since $\mathcal{X}(i) \neq \mathcal{X}(j)$, x_j appears in $T_{\mathcal{X}(i)}$ and hence, $x_j = 0,\ T_{\mathcal{X}(i)} = 0$ since all 3 terms are 0, the examples get label 0 under $T_R \vee T_B \vee T_G$. So it's consistent with all of the examples.

Given a 3-term DNF that's consistent with our constructed data set, let's (arbitrarily) label the terms $T_R^{'},\ T_B^{'},\ T_G^{'}$. I can construct a 3-coloring $\mathcal{X}^{'}$ as follows:
For the positive example corresponding to vertex i, $\mathcal{X}^{'}(i)$ be the label of some term that is satisfied on this positive example. $T_R^{'} \vee T_B^{'} \vee T_G^{'}$ consistent with the data, there must be some term that is satisfied so every vertex has at least a legal color.

Then we can see that $\forall (i,j) \in E,\ \mathcal{X}^{'}(i) \neq \mathcal{X}^{'}(j)$. Suppose $\mathcal{X}^{'}(i) = \mathcal{X}^{'}(j)$ for some $i \neq j$. Because of how $\mathcal{X}^{'}$ was chosen, the positive examples for $i \neq j$ both satisfy $T_{\mathcal{X}^{'}(i)}$. Since in vertex i's example $x_i = 0,\ so\ T_{\mathcal{X}^{'}(i)}$ cannot contain x_i. Since in vertex j's example $x_i = 1,\ so\ T_{\mathcal{X}^{'}(j)}$ cannot contain $\neg x_i$ either. Since the negative example for any pair(i, j) correspoding to an edge differ from the positive example for vertex j only in the ith coordinate, and we know that neither x_i nor $\neg x_i$ appear in their term, the term $T_{\mathcal{X}^{'}(i)}$ must still be satisfied, so the example for the edge(i, j) would receive label 1. But this is a negative exmple for every $(i,j) \in E$. Since we obtain a 3-term DNF, $T_R^{'} \vee T_B^{'} \vee T_G^{'}$ that is consistent with our constructed data set, this can't occur. Therefore, $\mathcal{X}^{'}$ is a proper 3-coloring of G.

3 Algorithm for improper learning of 3-term DNF

Claim: Every 3-term DNF can be written as a 3-term CNF.

3.1 Transformation from DNF to CNF

- 3-term DNF: OR of at most 3 ANDs of literals.
- 3-term CNF: AND of ORs of at most 3 literals each.

So by the defination of distributive law, we can transforme DNF into CNF as follows:

$$\begin{gathered}(l_{11} \wedge l_{12} ... \wedge l_{1k}) \vee T_2 \vee T_3 \\ \Rightarrow (l_{11} \vee T_2 \vee T_3) \wedge (l_{12} \vee T_2 \vee T_3) \wedge ... \wedge (l_{1k} \vee T_2 \vee T_3) \\ \Rightarrow (l_{11} \vee l_{12} \vee l_{13}) \wedge (l_{21} \vee l_{22} \vee l_{23}) \wedge ... \\ \Rightarrow \wedge_{l,l^{'},l^{''} \in T_1,T_2,T_3} (l \vee l^{'} \vee l^{''})\end{gathered}$$

3.2 3-term CNF

Theorem: 3 CNFs are (properly) PAC learnable.
Pf:
For each possible OR of at most 3 literals over n attributes, there's at most $(2n)^3$ such set, $O(n^3)$ of them. We'll construct a new attribute y_c.

$$(x_1, x_2, ..., x_n) \longmapsto (y_{c1}, y_{c2}, ..., y_{ck}) \text{ s.t. } \forall c_i, y_{c_i} = c_i(x_1, ...x_n)$$

Note the 3-term CNF $c_1 \wedge c_2 \wedge ... \wedge c_k$ in the original feature space is equivalent to the conjunction $y_{c_1} \wedge y_{c_2} \wedge ... \wedge y_{c_k}$ in this new feature space. I mean that our new data set is labeled by this conjunction. If I run our PAC-learning algorithm for conjunction on this new poly-larger data set, it still runs in poly time and it produces as output over the $\{y_{c_i}\}'s$. To see that upon substituing the $c_i(x_i, ..., x_n)$ formulas satisfactory prediction of the 3 CNF.

We notice the map that takes the x's to y's is a 1-1 correspondence, so if we consider the event that

$$\{y_{c_1} \wedge ... \wedge y_{c_k} \text{ is wrong on f(x)}\}$$

occurs with prob $< \varepsilon$ on f(D) which is the distribution over y's induced by D. Since $Pr_D[x] = Pr_{f(D)}[f(x)]$, we find that also the event

$$\{c_1 \wedge ... \wedge c_k \text{ is wrong on x}\}$$

has prob $< \varepsilon$ on D.

Properness: Modify the Elimination Algorithm to start with only the positive literals in h.