Recall:

**Lemma** If there is a proper PAC-learning algorithm for a class of representations $C$, then there is a randomized polynomial-time algorithm for solving $\text{Consis}^p(n) = \{([x^{(1)}, b^{(1)}], ..., [x^{(1)}, b^{(1)}]), c \in C, |c| \leq p(n) \forall i, c(x^{(i)} = b^{(i)})\}$ where $p(n)$ means polynomial in terms of number of attributes, $[x^{(1)}, b^{(1)}], ..., [x^{(1)}, b^{(1)}]$ is the input "labeled examples" (or training set), $c$ is the output, $|c|$ is the number of bits to represent $c$.

## 1 Hardness of proper learning of 3-Term DNF

### 3-Term DNF

3-Term DNF is a set of disjunctions $T_1 \lor T_2 \lor T_3$, where each $T_i$ is a conjunction of literals over the boolean variables $x_1, ..., x_n$. And it can be written down in $6n$ bits.

Our strategy to show $\text{Consis}^6_{3\text{-Term DNF}}$ is NP-complete is by stating

$\Rightarrow$ we don’t believe there is an efficient algorithm for $\text{Consis}^6_{3\text{-Term DNF}}$

is contrapositive of Lemma

$\Rightarrow$ there is no proper PAC-learning of 3-Term DNF

Our starting point is a famous NP-complete problem, 3-coloring:

$$3\text{-coloring} = \{(G, \mathcal{X}), \text{for every } (u, v) \in E, \mathcal{X}(u) \neq \mathcal{X}(v)\}$$

where $G = (V, E)$ is an $n$-vertex undirected graph, $\mathcal{X} : V \to R, G, B$ is a "proper coloring", i.e., for every $(u, v) \in E, \mathcal{X}(u) \neq \mathcal{X}(v)$

**Theorem** Unless NP has randomized polynomial time algorithms, there is no proper PAC-learning algorithm for 3-Term DNF.

**Proof** It suffices to show how to reduce 3-coloring to $\text{Consis}^6_{3\text{-Term DNF}}$ (following the earlier reasoning), i.e. gives a subroutine for $\text{Consis}^6_{3\text{-Term DNF}}$, we’ll show how to solve 3-coloring.

### 1.1 3-coloring problem

Given undirected graph $G = (V, E)$ as input, subroutine for $\text{Consis}^6_{3\text{-Term DNF}}$ show how to find a proper coloring $\mathcal{X}$ of $G$ if one exists. Set up some correspondence between the learning problem and 3-coloring, what do the attributes correspond to? What do the examples represent?

Given the $n$-node graph $G = (V, E)$ as input, we construct the following list of $n$-attributes examples:

- For each $i^{th}$ vertex, we have the example
  $$(1, 1, 1, ..., 0_{i^{th}}, 1, ..., 1, 1)$$
as positive example, where the only "0" appears at the \(i^{th}\) position

- For each edge \((i,j)\), we have the example

\[(1, 1, 1, \ldots, 0_{i^{th}}, 1, \ldots, 0_{j^{th}}, 1, \ldots, 1, 0)\]

as negative example, where the two "0"s appears at \(i^{th}\) and \(j^{th}\) position

We’ll show

1) When \(G\) has a proper 3-coloring, then is a consistent 3-form DNF

2) We can read a 3-coloring from any 3-Term DNF that labels these examples correctly.

Given some proper 3-coloring \(\mathcal{X}\) of \(G\), we’ll associate each of the three terms with one of the three colors; in the term \(T_R\) for color \(R\), we’ll include the literal \(\mathcal{X}_1\) for each node s.t. \(\mathcal{X}(i) \neq R\) similarly for \(T_B\) and \(T_G\)

1.2 Consistence analysis

To see that this is consistent:

For each positive example, the vertex \(i\) gets one of the three colors, the term corresponding to that color only contains literals \(x_j\) for \(j \neq i\); since these \(x_j = 1\) in the example for vertex \(i\), this term is satisfied, so the formula over all is also 1 ⇒ All positive examples are correct.

For each negative example, consider any \((i,j) \in E\), since \(\mathcal{X}\) is a proper 3-coloring, the end points satisfy \(\mathcal{X}(i) \neq \mathcal{X}(j)\). So in the term for \(\mathcal{X}(i)\), the literal \(x_j\) appears, so this term is 0 since \(x_j = 0\) in this example. Similarly, the term for color \(\mathcal{X}(j)\), the literal \(x_i=0\) appears, making the term 0. In the third term, both \(x_i = 0\) and \(x_j = 0\) appear. Since all three terms are false, the formula is 0. ⇒ All negative examples are correct.

⇒ All examples are labeled correct.

Now, suppose we got a 3-Term DNF form our subroutine for Consis\(_3^n\) 3-Term DNF for this set of examples. Arbitrarily associate the terms to colors, for each \(i^{th}\) vertex, we assign \(\mathcal{X}(i)\) equal to the color of some term that is satisfied on the example for the \(i^{th}\) vertex. Since this DNF solved Consis\(_3^n\) 3-Term DNF, some such term must exist.

Remains to show that this coloring is proper.

Consider any two vertices \(i\) and \(j\) that receive the same color. It’s enough to show that there is no edge between \(i\) and \(j\). These vertices satisfy the same term. and \(x_i\) is false in the example for \(j\), true in the positive for \(j\), and the term is true in both cases. Neither \(x_i\) nor \(\neg x_i\) can appear in this term.

Notice: the only difference between the positive example for \(j\), and the (hypothetical) example for an edge \((i,j)\) is that the \(i^{th}\) attribute is 1 in the positive example, and 0 in the negative (edge) example. But since this term includes neither \(x_i\) nor \(\neg x_i\), it must still be satisfied on an example that we would have constructed for an edge \((i,j)\), which would give that term a positive label. Since a solution to Consis\(_3^n\) 3-Term DNF would have to be labeled that example 0 if it had been included, so such example could have been included, so there is no edge \((i,j)\) in \(G\). Easy to compute. ◇
2 Algorithm for improper learning of 3-Term DNF

2.1 Relation between 3-CNF and 3-Term DNF

**Claim** every 3-Term DNF can be written as a 3-CNF (AND of ORs of at most 3 literals)
(We'll show how to properly learn 3-CNF ⇒ improper learning of 3-Term DNF)

We have Distribute law as:

\[(A \land B) \lor C \equiv (A \land C) \lor (B \land C)\]
\[(A \lor B) \land C \equiv (A \lor C) \land (B \lor C)\]

Then we can distribute 3-Term DNF as:

\[(l^{(1)}_1 \land l^{(1)}_2 \land ... \land l^{(1)}_{k_1}) \lor (T_2 \lor T_3) \]
\[\Rightarrow (l^{(1)}_1 \lor T_2 \lor T_3) \land (l^{(2)}_2 \lor T_2 \lor T_3) \land ... \land (l^{(1)}_{k_1} \lor T_2 \lor T_3) \]
\[\Rightarrow (l^{(1)}_1 \lor l^{(2)}_2 \lor l^{(3)}_{k_1}) \land ... \land (l^{(1)}_{k_1} \lor l^{(2)}_{k_2} \lor l^{(3)}_{k_3}) \]

Note: max size of any 3-CNF is \((\frac{2n}{3})\) since that is the number of possible terms.

2.2 Proper learning of 3-CNF

**Theorem** 3-CNFs are properly PAC-learnable.

**Proof** We again transform the "feature space". We'll create an attribute for each of the \((\frac{2n}{3})\) clauses (ORs); our 3-CNF is only a conjunction over the clauses. We run Elimination on this transformed data. Given an example \(x \in (0, 1)^n\), we set the attribute \(y_c\) for a clause \(c\) equal to \(C(x)\).

Notice: for any 3-CNF \(C_1 \land C_2 \land ... \land C_k\),
\n\[(C_1 \land C_2 \land ... \land C_k)(x) = y_1 \land y_2 \land ... \land y_k\]

So

(1) if the data was labeled by a 3-CNF, the transformed data will be labeled by the corresponding conjunction

(2) any conjunction on the transformed data computes the label correctly on \((y, b)\), if and only if the corresponding 3-CNF computes the label correctly on \((x, b)\)

(We'll modify Elimination to only use positive literals, the analysis remains the same.)

By the correctness of elimination and (1), we are guaranteed to find a conjunction such that

\[\text{Pr}_y[y_1 \land y_2 \land ... \land y_k \neq b] < \epsilon\]

by (2),

\[\Rightarrow \text{Pr}_x[y_1 \land y_2 \land ... \land y_k \neq b] < \epsilon\]

Thus, the 3-CNF we read off \(C_1 \land C_2 \land ... \land C_k\) is satisfactory with probability \(1 - \delta\) by the analysis of Elimination \(\diamond\)
2.3 Improper learning of 3-Term DNF

**Definition** For classes of representation \( \mathcal{C} = \cup_n \mathcal{C}_n \) and \( \mathcal{H} = \cup_n \mathcal{H}_n \) s.t. \( \mathcal{C}_n \) and \( \mathcal{H}_n \) are representations over \( \mathcal{X} \). We say that \( \mathcal{C} \) is **efficiently PAC-learnable using** \( \mathcal{H} \) if there is an algorithm that, given access to examples from \( \mathcal{X}_n \) for some \( n \in \mathbb{N} \), and given as input \( n, \epsilon \in (0, 1/2), \delta \in (0, 1/2) \) and \( \text{size}(c) \) for some \( c \in \mathcal{C}_n \) such the labels are computed by \( c \), runs in time polynomial in \( n, 1/\epsilon, 1/\delta, \text{size}(c) \) and with probability \( 1 - \delta \) returns \( h \in \mathcal{H} \) such that \( \Pr[h(x) = c(x)] \geq 1 - \epsilon \)

Today’s example, 3-Term DNF is PAC-learnable using 3-CNF but not using 3-Term DNF. For a fixed representation class \( \mathcal{H} \) any \( \mathcal{C}' \subseteq \mathcal{C} \) is learnable using \( \mathcal{H} \) if \( \mathcal{C} \) is.

![Figure 1: Relation between certain circuit](image)

**Figure 1:** Relation between certain circuit