Learning Horn Expressions using Queries

1 Horn Expressions

Definition: CNF where each disjunction has at most one positive literal. Since each disjunction can be written as a rule, e.g. \( \bar{a} \lor \bar{b} \lor \bar{c} \lor d = abc \rightarrow d \), Horn expressions are often written as conjunctions of rules. For example: \( f = (x_2x_3 \rightarrow x_5) \land (x_4 \rightarrow x_5) \).

The main property of Horn Expressions is: \( X \models f \) and \( Y \models f \), then \( X \cap Y \models f \). This is equivalent to: if \( X \cap Y \not\models f \), then \( X \not\models f \) or \( Y \not\models f \).

2 Learning Model

Equivalence Query(EQ): Learner presents hypothesis, oracle says yes or no and if the answer is no, it provides counter example.

Membership Query(MQ): Learner presents example \( x \), oracle gives label(\( x \)).

3 Learning Horn Expressions using Queries

Some ideas:

Suppose we are given a negative example: 01110110, since this example violates some rule, this gives some information. In particular, we can construct the following rules:

\[
\begin{align*}
x_2x_3x_4x_6x_7 & \rightarrow x_1 \\
x_2x_3x_4x_6x_7 & \rightarrow x_5 \\
x_2x_3x_4x_6x_7 & \rightarrow x_8
\end{align*}
\]

We know one of these is correct, but they are perhaps too specific. Call this operation taking an example \( x \) and producing a set of rules \( rules(x) \), and extend the operation to sets of examples by applying it to each example in the set.

Now if we are given a positive example, say: 01111110, then we know that any rules it violates is wrong. In particular, \( x_2x_3x_4x_6x_7 \rightarrow x_1 \) and \( x_2x_3x_4x_6x_7 \rightarrow x_8 \) are wrong. Therefore we hope to get such positive examples as positive counter examples.

Using these ideas we can construct the following learning algorithm.
Algorithm to Learn Horn Expressions:
1. $S = \emptyset$ where $S$ is a sequence of examples.
2. $R = \text{rules}(S)$.
3. Ask EQ(R).
    3.1 If given PCEX (positive counter example) $X$: $X \models \text{target}, X \not\models R$. Remove rules in $R$ violated by $X$ and go to step 3.
    3.2 If given NCEX (negative counter example) $X$: $X \not\models \text{target}, X \models R$. let $|S| = k$.
        For $i = 1$ to $k$
        
        If $|S_i \cap X| < |S_i|$ and MQ($S_i \cap X$) is negative
        
        Replace $S_i$ by $S_i \cap X$ and quit the loop.

        If no $S_i$ was replaced, then add $X$ as last element of $S$ and go to step 2.

We say that an example covers a rule if it satisfies the condition (and may or may not satisfy the conclusion). Notice that if an example does not cover a rule, then if we flip some of its 1 bits to 0 it will still not cover and not falsify the rule. The bitwise intersection of two examples covering the same rule still covers the rule. We therefore have the following fact.

**Fact:** If $X$ cover rule $c$ and $Y$ violates $c$ then $X \cap Y$ violates $c$.

**Lemma 1:** If NCEX $X$ violates clause $c \in \text{target}$ and $S_i$ covers $c$, then algorithm replaces $S_j$ for some $j \leq i$.

**Lemma 2:** $\forall i, k$ where $i < k$, if $S_k$ violates $c$, then $S_i$ does not cover.

**Theorem:** Algorithm learns Horn Expression with $\leq nm + mn \cdot mn$ EQs where target has $m$ clauses and $\leq nm \cdot m$ MQs.

**Proof of Theorem:** By Lemma 2, every clause of the target is violated by at most one element of $S$, and therefore $|S| \leq m$. Since $S_i$ is replaced only if $|S_i|$ is strictly smaller, it is refined at most $n$ times. Therefore we have $\leq nm$ NCEX.

After each NCEX, we generate $\leq mn$ rules, therefore we have $\leq mn \cdot mn$ rules to remove, that is $\leq m^2 n^2$ PCEX. Since every NCEX requires $\leq m$ MQs, in total we have $\leq nm \cdot m$ MQs.

**Proof of Lemma 1:**

We want to show that if the algorithm did not replace $S_j$ where $j < i$, then it surely replaces $S_i$. For this proof, we need to show:

1. $S_i \cap X$ is negative. This is clearly true by the fact given above since $X$ violates the rule and $S_i$ covers it.
2. $|S_i \cap X| < |S_i|$

We know for sure that $|S_i \cap X| \leq |S_i|$. Assume part 2 does not hold, then we have $|S_i \cap X| = |S_i|$, in this case the "1" bits in $X$ are a superset of the "1" bits in $S_i$.

Now consider consequent($c$), let’s call it $y$:

If $y$ is false in $S_i$, then the rule $\text{ones}(S_i) \rightarrow y$ is correct, thus it is not removed from the hypothesis and $X$ violates it. This $X$ can not be a counter example, therefore we have a contradiction here.
If $y$ is true in $S_i$, then we have $y$ is true in $S_i$ but false in $S_i \cap X$. Therefore we have $|S_i \cap X| < |S_i|$. Again this contradicts our assumption.

**Proof of Lemma 2:** not covered in class.