The VC Dimension and Compression Algorithm

1 Vapnik Chervonenkis Dimension

Let $X$ be an instance space, $S \subseteq X$, $C$ is a concept class. Define:

$$\Pi_C(S) = \{c \cap S | c \in C\}$$  \hspace{1cm} (1)

i.e. the set of subsets of $S$ that can be obtained using $c \in C$

$$\Pi_C(m) = \max_{S: |S|=m} |\Pi_C(S)|$$  \hspace{1cm} (2)

We define $C$ shatters $S$ if $|\Pi_C(S)| = 2^{|S|}$, i.e. get all possible behaviors. The Vapnik-Chervonenkis Dimension of $C$:

$$VCD(C) = \max_{S: S \text{ shatters } S} |S|.$$  

If $C$ can shatter arbitrarily large sets, we say that $VCD(C) = \infty$.

To prove $VCD(C) = d$, we need to show:

1. There is at least one set of size $d$ that is shattered.
2. No set of $d+1$ points is shattered.

Example 1: $VCD(\text{Intervals}) = 2$: we can see that a labeling "+ + +" of 3 points can not be obtained by any single interval. On the other hand, any two points can be shattered since all labelings: "- -", "+ +", "+ +" can be easily obtained.

Example 2: $VCD(\text{Union of two Intervals})$, here the concept= $[a, b] \cup [c, d]$:

(1) Pick 4 points, we can simply shatter the "left" 2 points with one interval, and the "right" 2 points with one interval to get any labeling of the 4 points.

(2) For 5 points, we can see that "+ + + + +" can not be obtained by any two intervals.

Example 3: $VCD(\text{Union of } k \text{ Intervals})$, here the concept= $[a_1, b_1] \cup \cdots \cup [a_k, b_k]$. We can see that similar to the argument for $k = 2$, the pattern "+ + + + +" cannot be realized and it is easy to shatter $2k$ points in parts.

Example 4: $VCD(\text{axis parallel rectangle})$. In class we showed that any labeling of some 4 point configuration (for example the set $(0, 2), (1, 0), (2, 3), (3, 1))$ can be obtained. For any five points, we can pick 4 points that together include the minimum and maximum $x$ and $y$ coordinates. Give the 4 points with these extremal values label + and the fifth one label -. Then there is no concept that can obtain the labeling.
Example 5: \( VCD(\text{Conjunctions}) = n \) where \( n \) is the number of features in the domain, and we assume empty conjunction is true on every example. Here we only prove \( VCD(\text{Conjunctions}) \geq n \). The other direction appears in the homework.

Let \( O_i \) = string of all 1’s except in \( i \)’th position. We define \( S = \{O_i\}^n_{i=1} \). For example if \( n = 5 \) we have 01111, 10111, 11011, 11101, 11110. We next show that \( S \) is shattered.

Recall that the only zero bit in \( O_i \) is \( x_i \). It is easy to see that if \( i \in R \), \( O_i \) is a negative example for \( c \) since \( x_i \) is in the conjunction; otherwise, \( O_i \) is positive since \( x_i \) is not in conjunction. Since the choice of labeling was arbitrary we have shown that all labelings can be obtained and \( S \) is shattered.

2 Compression Algorithm

We have so far seen one method to obtain generic convergence bounds for PAC learning via Occam algorithms for finite hypotheses classes or when the VCD is finite. Compression algorithms give an alternative method that does not depend on such properties of the concept class (but instead depends on the existence of an algorithm).

Fix any learning algorithm \( L \) that takes an ordered sample \( B \) as input and produces a hypothesis \( hyp(B) \) based on the sample.

Algorithm \( A \) is a compression algorithm for concept class \( C \) (w.r.t \( L \)) with size \( d \) if when given ordered samples \( S = \{x_1, \ldots, x_m\} \), \( A \) outputs \( B \subset S \) s.t.

1. \(|B| \leq d\).
2. \( hyp(B) \) is consistent with \( S \).

**Theorem:** A compression scheme is a PAC learner with \( \epsilon = \frac{1}{m-d} \left[d \log \left(\frac{em}{d} \right) + \log \frac{1}{\delta}\right] \).

**Proof:** Consider an index set \( T \subseteq \{1, \ldots, m\} \) where \(|T| = k \leq d\) and define \( S|_T = \text{subset of } S \text{ with examples at indices } T \).

Define

\[
\text{consistent}(T) = hyp(S|_T) \text{ is consistent with } S
\]
\[
\text{bad}(T) = \text{err}(hyp(S|_T)) > \epsilon
\]

Note that for a fixed index set \( T \), the subsamples \( S|_T \) and \( (S \setminus S|_T) \) are independent. When algorithm fails, it chooses \( T \) with \( \text{consistent}(T) \cap \text{bad}(T) \).

\[
Pr(\text{consistent}(T) \cap \text{bad}(T)) = Pr(\text{bad}(T)) \cdot Pr(\text{consistent}(T)|\text{bad}(T))
\]
\[
\leq Pr(\text{consistent}(T)|\text{bad}(T))
\]
\[
\leq (1 - \epsilon)^{m-|T|}
\]

We therefore have

\[
Pr(\text{Compression Algorithm outputs } hyp \text{ with error } > \epsilon)
\]
\[
= Pr(\text{Algorithm picks } T \text{ with } \text{consistent}(T) \cap \text{bad}(T))
\]
\[ \leq \bigcup_{T} Pr(\text{consistent}(T) \cap \text{bad}(T)) \]
\[ \leq \text{via union bound and bound in previous formula} \]
\[ \sum_{k=0}^{d} \binom{m}{k} (1 - \epsilon)^{m-k} \]
\[ \leq (1 - \epsilon)^{m-d} \sum_{k=0}^{d} \binom{m}{k} \]
\[ \leq (1 - \epsilon)^{m-d} \left( \frac{em}{d} \right)^{d} \]
\[ \leq e^{-\epsilon(m-d)} \left( \frac{em}{d} \right)^{d} \leq \delta \]

Here we use the inequality \( \sum_{k=0}^{d} \binom{m}{k} \leq \left( \frac{em}{d} \right)^{d} \). Solving the last inequality for \( \epsilon \) gives the bound in the theorem.

Note that we can also solve for \( m \) to get bounds as in previous theorems where \( m \geq \left( \frac{2d}{\epsilon} \right)^{2} \ln \frac{1}{\delta} \) is easy to derive (but we can reduce the quadratic dependence).