Computational Learning Theory: Part 2

Timothy Hahn

Montana State University

April 9, 2008

hahn@cs.montana.edu

7.4.1 Shattering a Set of Instances

- The Vapnik-Chervonekis (VC) dimmensions are based on the complexity of H, not on the number of distinct hypotheses $|H|$.
- Consider some subset of instances $S \subseteq H$. Each hypothesis h from H partitions S into two subsets $\{x \in S \mid h(x) = 1\}$ and $\{x \in S \mid h(x) = 0\}$.

Definition

A set of instances S is **shattered** by hypothesis space H if and only if for every dichotomy of S there exists some hypothesis in H consistent with this dichotomy.

- Given some instance set S, there are $2^{|S|}$ possible dichotomies.

7.4.2 Vapnik-Chervonekis Dimension

- An unbiased hypothesis space H is one that shatters the instance space X.
- The larger the subset of X that can be shattered, the more expressive H.

Definition

The **Vapnik-Chervonenkis dimension**, $VC(H)$, of hypothesis space H defined over instance space X is the size of the largest finite subset of X shattered by H. If arbitrarily large finite sets of X can be shattered by H, then $VC(H) \equiv \infty$.

- For any finite H, $VC(H) \leq log_2|H|$.

7.4.2.1 Illustrative Examples

Example 1: Height

- Suppose the instance space X is the set of real numbers describing the height of people.
- Suppose the hypothesis space H is the set of intervals on the real number line of the form $a < x < b$ where a and b are any real constants.
- What is the $VC(H)$?
- Note that even though H is infinite, $VC(H)$ is finite.

7.4.2.1 Illustrative Examples

Example 2: Points on the x, y Plane

- Suppose the instance space X is the set of instances corresonding to points on the x, y plane.
- Suppose the hypothesis space H is the set of all linear decision surfaces in the plane. Note that H is also the hypothesis space correspoding to a single perceptron unit with two inputs.
- Two distinct points can be clearly shattered by H.
- Three distinct points (assuming they are not colinear) can be shattered by H.
- What is the $VC(H)$?

7.4.2.1 Illustrative Examples

Example 3: Boolean Literals

- Suppose the instance space X is the set of statements formed by the conjunction of exactly three boolean literals.
- Suppose the hypothesis space H is the set of statements formed by the conjunction of up to three boolean literals.
- What is the $VC(H)$?

7.4.3 Sample Complexity and the VC Dimension

- Upper bound on number of training examples.

$$m \geq \frac{1}{\epsilon}(4log_2\frac{2}{\delta} + 8VC(H)log_2\frac{13}{\epsilon})$$

- Lower bound on number of training examples.

$$max[\frac{1}{\epsilon}log_2\frac{1}{\delta}, \frac{VC(C)-1}{32\epsilon}]$$

- Note that the upper bound is based upon $VC(H)$ while the lower bound is based upon $VC(C)$. Why?

7.4.4 VC Dimension for Neural Networks

Theorem

Let G be a layered directed acyclic graph with n input nodes and $s \geq 2$ internal nodes, each having at most r inputs. Let C be a concept class over $\mathcal{R}^r$ of VC dimension d, corresponding to the set of functions that can be described by each of the s internal nodes. Let C_G be the G-composition of C, corresponding to the set of functions that can be represented by G. Then $VC(C_G) \leq 2(ds)log_2(es)$, where e is the base of the natural logarithm.

7.4.4 VC Dimension for Neural Networks

- We can substitue this inequality into equation (7.7):

$$m \geq \frac{1}{\epsilon}(4log\frac{2}{\delta} + 16(r+1)slog(es)log\frac{13}{\epsilon})$$

- The result is not directly applicable to networks trained with BACKPROPAGATION networks.
- The result applies to networks of perceptrons, not *sigmoid units*.
- The result does not account for the fact that BACKPROPAGATION trains a network by beginning with near-zero weights then iteratively modifying these weights. This inductive bias towards near-zero weights is not captured by this formula.

7.5 Mistake Bound Model

- In the *mistake bound* model of learning, the learner is evaluated by the total number of mistakes it makes before it converges to the correct hypothesis.
- We assume the learner receives a sequence of training examples; however, we demand that upon receiving each example x, the learner must predict the target value $c(x)$ before it is shown the correct value.
- The question considered is "How many *mistakes* will the learner make in its predictions before it learns the target concept?"

7.5.1 Mistake Bound for the FIND-S Algorithm

FIND-S Algorithm

- Initialize h to the most specific hypothesis $l_1 \wedge !l_1 \wedge l_2 \wedge !l_2 ... l_n \wedge !l_n$.
- For each positive training instance x remove from h any literal that is not satisfied by x.
- Output hypothesis h.

7.5.1 Mistake Bound for the FIND-S Algorithm

- FIND-S will never classify a negative example as positive, h is always at least as specific as c.
- Maximum number of mistakes is $n + 1$ in the case where $(\forall x)c(x) = 1$ and a sequence of instances where one literal is removed per mistake.

7.5.2 Mistake Bound for the HALVING Algorithm

- Consider an algorithm that learns by maintaining a description of the version space, such as CANIDATE ELIMINATION and LIST-THEN-ELIMINATE from Chapter 2.
- Assume prediction is made using a majority vote among the hypotheses in the current version space.
- Often called the HALVING Algorithm.

7.5.2 Mistake Bound for the HALVING Algorithm

- Note that the algorithm makes a mistake only when the majority incorrectly classifies an example. At this time, the version space will be reduced to at most half.
- Maximum number of mistakes before the version size is equal to one is $log_2|H|$.
- Example: When $|H| = 7$, first mistake reduces $|H|$ to 3 and the next mistake will reduce $|H|$ to 1.
- This is a worst case bound.

7.5.3 Optimal Mistake Bounds

- Goal: Find the optimal mistake bound over all possible learning algorithms.

Theorem

Let C be an arbitrary nonempty concept class. The **optimal mistake bound** *for C, denoted $Opt(C)$, is the minimum over all possible learning algorithms A of $M_A(C)$.*

- Interesting relationship:
  $VC(C) \leq Opt(C) \leq M_{Halving}(C) \leq log_2(|C|)$

7.5.4 Weighted-Majority Algorithm

- The algorithm begins by assigning a weight of 1 to each prediction algorithm and then considers training examples.
- When an algorithm misclassifies a new training example, its weight is decreased by multiplying it by some number β where $0 \leq \beta < 1$.
- When $\beta = 0$, Weighted-Majority is identical to Halving.

7.5.4 Weighted-Majority Algorithm

Theorem

Let D be any sequence of traning examples, let A be any set of n prediction algorithms, and let k be the minimum number of mistakes made by any algorithm in A for the training sequence D. Then the number of mistakes over D made by the Weighted-Majority *algorithm using $\beta = \frac{1}{2}$ is at most $2.4(k + log_2 n)$*

- The number of mistakes made by the Weighted-Majority algorithm will never be greater than a constant factor times the number of mistakes made by the best member of the pool plus a term that grows logarithmically times the size of the pool.