NIPS 2000 Workshop

John Langford

October 15, 2000

1 Introduction

I present a summary of various bounds which are of interest to the workshop. Some of these bounds appeared in "Beating the Holdout" by Avrim Blum, Adam Kalai, and John Langford[1]. Others are unpublished or work-in-progress results which I would like to discuss.

2 Terminology and Definitions

(principally borrowed from the "Beating the Holdout" paper)

In many situations, a learning algorithm must simultaneously produce a hypothesis having low generalization error and a high-accuracy estimate of this error. We make the usual assumption that data is drawn independently from some fixed distribution, and the error of a hypothesis on examples from this distribution is called the generalization error or *true error*.

Several procedures exist for generating pairs of the form $\langle$hypothesis, estimated error$\rangle$. Such a procedure can be scored in two dimensions: the true error of its hypothesis, and the *error discrepancy*: $|$(estimated error) $-$ (true error)$|$. Unfortunately it is difficult to predict the true error directly so we will focus on bounding the error discrepancy.

2.1 Basic Definitions

Let X be the instance space and let $\mathcal{D}$ be a fixed distribution over X. We also assume a fixed target function $f : X \longrightarrow \{0,1\}$. A learning algorithm produces a hypothesis $h : X \longrightarrow [0,1]$. We allow the range to be $[0,1]$, for convenience, so that we have a notion of averaging hypotheses. The error of this hypothesis on a particular example $x \in X$ is $e_h(x) = |h(x) - f(x)|$. The *true error* of this hypothesis is $\bar{e}_h = E_{x\in\mathcal{D}}[e_h(x)]$. The *training error* of a hypothesis is $\hat{e}_h = E_{x\in S} e_h(x)$.

2.2 Training Procedures

The *resubstitution procedure* generates a hypothesis by training on all the data and generates an error estimate by measuring the number of mistakes of the learned hypothesis on the same data used for training. The number of examples used in the resubstitution procedure will be m_{train} and the empirical error will be denotes as: $\hat{e}_{train}$.

The *hold-out procedure* divides the data in two parts: the training set, on which the hypothesis is trained, and the hold-out set, on which its performance is measured. Among the nice properties that this procedure obeys are Hoeffding bounds guaranteeing, regardless of the learning algorithm, that with high probability the error discrepancy will be small. The number of examples will be denoted as m_{test} and the empirical error as $\hat{e}_{test}$.

The *k-fold procedure* divides the data into k equally sized *folds*. It then produces a hypothesis by training on $k-1$ folds and testing on the remaining fold. This is repeated for each fold, and the observed errors are averaged to form the *k-fold estimate*. It is not obvious what hypothesis to output along with this error estimate. In previous analysis, the final hypothesis was often a new hypothesis trained on all the data. Because this hypothesis was constructed using more data than the hypotheses used for computing the error estimate, in order to argue for the accuracy of the estimate one needs some assumption that limits the effect of this extra training data. In particular, previous work gives *sanity-check bounds[3]* which show that the k-fold estimate is almost as good as the training error estimate in the resubstitution procedure, under the assumption that the learning algorithm has some form of hypothesis stability.

Our k-fold procedure, instead, outputs the *k-fold hypothesis*, a meta-hypothesis that, given an example x, randomly chooses one of the k generated hypotheses h_i and outputs the prediction of that hypothesis, $h_i(x)$. Alternatively, if hypotheses are allowed to make predictions in $[0,1]$ and we are using L_1 loss, this is equivalent, in terms of its true error, to outputting the average value of $h_1(x), \ldots, h_k(x)$ when the true labels are $0, 1$. The empirical test errors of each hypothesis will be denoted as $\hat{e}_1, ...\hat{e}_k$ with the average test error as $\hat{e}_{cv}$ and the average true error as $\bar{e}_{cv}$.

The *progressive validation procedure*, like the hold-out procedure, first selects s examples for testing, and the remainder are for training only. It generates a sequence of s hypotheses, where the ith hypothesis is trained on all of training data *plus* the first $i-1$ examples of the test set, and tested on the ith example of the test set. Repeating this for $1 \leq i \leq s$, we count the number of mistakes to produce an error estimate. The hypothesis returned, as above, is a meta hypothesis which randomly selects among the s generated hypotheses to make its prediction. Let $\hat{e}_{pv}$ and $\bar{e}_{pv}$ be the average test error and average true error over each of the chosen hypotheses.

3 Previous Results

This is a quick summary of previous results for bounding the error discrepancy of the various procedures.

Resubstitution

$$\Pr(\bar{e}_{train} - \hat{e}_{train} > \epsilon) \leq |H| e^{-2 m_{train} \epsilon^2}$$

This is a folk-bound derivable directly from the Chernoff or Hoeffding bound.

Hold-out

$$\Pr(\bar{e}_{test} - \hat{e}_{test} > \epsilon) \leq e^{-2 m_{test} \epsilon^2}$$

This is simply the Chernoff bound in the setting of machine learning.

K-fold

$$\Pr(\bar{e}_{cv} - \hat{e}_{cv} > \epsilon) \leq e^{-2 \frac{m}{k} \epsilon^2}$$

The proof follows from linearity expectation plus the hold-out bound. In "Beating the Holdout" we also proved that under a reasonable assumption there was a strict improvement on the hold-out procedure with $m_{test} = \frac{m}{k}$.

Progressive Validation

$$\Pr(\bar{e}_{pv} - \hat{e}_{pv} > \epsilon) \leq e^{-2 m_{test} \epsilon^2}$$

The proof is a little tricky. The interesting thing about this result is that we get a bound just as tight as the hold-out procedure but also may use an expected half of the held out data in our learned hypothesis.

4 Newer results

(all unpublished, and still work in progress)

4.1 Combined Resubstitution and Hold-out

Under most real world settings the hold-out bound is much tighter then the resubstitution bound. However, this is not always the case, especially when the quantity of data is so limited that holding out any would greatly increase the hypothesis error. It would be interesting to have a bound which uses information from both sources. This new bound would hopefully unify the resubstitution bound (commonly worked on in the COLT community) with the hold-out bound which is the standard practice of machine learning community. Ideally, there would be a bound which uses both pieces of information to arrive at a bound which is better then either individually.

There are three approaches to find this bound.

conjunction approach We can first try to work with the bound:

$$\Pr(\bar{e}_{train} - \hat{e}_{train} > \epsilon_{train} \wedge \bar{e}_{train} - \hat{e}_{test} > \epsilon_{test})$$

$$\leq \max(1, |H|e^{-2m_{train}\epsilon_{train}^2}) * e^{-2m_{test}\epsilon_{test}^2}$$

This bound is valid for a given ϵ_{train} and ϵ_{test} because the training set and test set are not dependent. Unfortunately, application of this bound is difficult. In practice, it is desirable to set the probability of failure to some constant (statistician like 0.05 but machine learning people might prefer 0.5) then solve for the ϵ which has this constant probability of failure. The total probability of failure is dependent on both ϵ_{test} and ϵ_{train}. It isn't possible to invert a function from $R^2 \to R$ so we must make some assumption to force the domain to be R. One natural assumption is that $\delta^{\frac{1}{2}} = |H|e^{-2m_{train}\epsilon_{train}^2}$ and $\delta^{\frac{1}{2}} = e^{-2m_{test}\epsilon_{test}^2}$. From here we can invert to find a bound on the true error which holds with high probability. The result: We have a bound which is slightly better than the worst of (resubstitution bound, hold out bound).

disjunction approach We can first try to work with the bound:

$$\Pr(\bar{e}_{train} - \hat{e}_{train} > \epsilon_{train} \vee \bar{e}_{train} - \hat{e}_{test} > \epsilon_{test})$$

$$\leq \max(1, |H|e^{-2m_{train}\epsilon_{train}^2}) + e^{-2m_{test}\epsilon_{test}^2}$$

We can derive this bound with an application of the inequality: $\Pr(A \vee B) \leq \Pr(A) + \Pr(B)$ to the resubstitution and holdout bounds. Once again, before inverting the bound, we must split our confidence δ, this time as $\frac{\delta}{2} = |H|e^{-2m_{train}\epsilon_{train}^2}$ and $\frac{\delta}{2} = e^{-2m_{test}\epsilon_{test}^2}$. And the result is: We have a bound which is slightly worse then the best of the two bounds.

generalized approach For any true error, the bounds for the holdout set and training set imply worst-case (largest error discrepancy) cumulative distributions $F_{train}(\hat{e}_{train})$ and $F_{test}(\hat{e}_{test})$ for the training and test set errors. We can then partition the input $(\hat{e}_{train}, \hat{e}_{test})$ space into two sets according to the test: $F_{train}(\hat{e}_{train})F_{test}(\hat{e}_{test}) > k$. The interesting thing about this partition is that the sets grow (or shrink) monotonocally with changing true error. By calculating the (worst case) probability of these sets we can arrive at a new bound. This new bound has the property that it can be tighter than either the resubstitution bound or the hold-out bound and is always not much worse then the best.

4.2 Tighter K-fold cross validation error

The "Beating the holdout" paper proves that under a reasonable assumption, the K-fold estimate is strictly tighter then the hold-out estimate. Empirically, we know that the K-fold cross validation error is significantly better than the hold-out error. This implies that there is a large gap between what we know empirically and what we know theoretically. It would be nice to close this gap.

For example, if we happened to observe that the every hypothesis chosen in the K-fold estimate was the same hypothesis then we would have the following (very tight) bound on the error discrepancy:

$$\Pr(\bar{e}_{cv} - \hat{e}_{cv} > \epsilon) \leq e^{-2m\epsilon^2}$$

Of course, it is rather rare to observe that every hypothesis is the same. We can attempt to relax the constraint of this event in two natural ways:

1. What if many (but not all) of the hypotheses are the same?
2. What if the hypotheses are different but agree on all or most of our examples?

Both relaxations significantly worsen the resulting bound. Relaxation (1) results in a bound which has variance dominated by the disagreeing set. For relaxation (2), let $\bar{e}_{h_1 h_2} = P_D(h_1(x) \neq h_2(x))$ and $\hat{e}_{h_1 h_2} = P_S(h_1(x) \neq h_2(x))$ we can show:

$$\Pr(\bar{e}_{h_1 h_2} > \hat{e}_{h_1 h_2} + \epsilon) \leq e^{-2\epsilon^2 m}$$

which is a bound on the probability that two hypothesis which differ a $\bar{e}_{h_1 h_2}$ portion of the time differ an observed $\hat{e}_{h_1 h_2}$ portion of the time or less. Looking at the worst-case difference over the hypothesis set, we get:

$$\Pr(\exists h_1, h_2 \in H : \bar{e}_{h_1 h_2} > \hat{e}_{h_1 h_2} + \epsilon) \leq \frac{1}{2}|H|(|H|+1)e^{-2\epsilon^2 m}$$

Given this bound on the disagreement between two hypotheses we can derive a bound on $\bar{e}_{CV}$ using techniques from Devroye and Wagner [2].

5 A Last Important Detail

It is worth mentioning that anyone using the above bounds on real data is using the wrong bound. The basic difficulty is that the underlying bound on coin flips:

$$\Pr(p - \hat{p} > \epsilon) \leq e^{-2m\epsilon^2}$$

is unnecessarily loose when p is near 0 or 1 as we often hope for in a machine learning setting. Instead, either direct calculation of binomial coefficients or the following bound (also due to Chernoff) should be used:

$$\Pr(p - \hat{p} > \epsilon) \leq e^{-mD(p-\epsilon||p)}$$

Here, $D(q||p)$ is the KL-divergence between the distribution of a coin with bias q and a coin with bias p.

References

[1] Avrim Blum, Adam Kalai and John Langford, "Beating the Hold-out, Bounds for KFold and Progressive Cross-Validation." Colt 1999, http://www.cs.cmu.edu/~jcl/papers/progressive_validation/coltfinal.ps

[2] L. P. Devroye and T.J. Wagner "Distribution-free performance bounds for potential function rules. IEEE transactions on Information Theory, IT-25(5):601-604, 1979

[3] M.J. Kearns and D. Ron "Algorithmic stability and sanity-check bounds for leave-one-out cross-validation" Colt 1997, http://www.research.att.com/~mkearns/papers/multi.ps.Z