ON OVERFITTING AVOIDANCE AS BIAS

by

David H. Wolpert

The Santa Fe Institute, 1660 Old Pecos Trail, Suite A, Santa Fe, NM, 87501, dhw@santafe.edu

SFI TR 92-03-5001

Abstract: In supervised learning it is commonly believed that penalizing complex functions helps one avoid "overfitting" functions to data, and therefore improves generalization. It is also commonly believed that cross-validation is an effective way to choose amongst algorithms for fitting functions to data. In a recent paper, Schaffer (1993) presents experimental evidence disputing these claims. The current paper consists of a formal analysis of these contentions of Schaffer's. It proves that his contentions are valid, although some of his experiments must be interpreted with caution.

Keywords: overfitting avoidance, cross-validation, decision tree pruning, inductive bias, extended Bayesian analysis, uniform priors.

1. Introduction

In his paper "Overfitting avoidance as bias" (1993), as well as some previous papers (1992a, 1992b), Schaffer disputes the common claim that trying to avoid overfitting by introducing a complexity cost necessarily improves generalization. That paper also presents evidence that it is possible that using cross-validation (Efron, 1979; Stone, 1974, 1977a, 1977b) to choose amongst learning algorithms can lead to worse generalization, despite the common lore to the contrary. Both arguments are based, for the most part, on numerical experiments, involving algorithms which fit decision trees to training sets ((Buntine, 1992) and references therein).

The current paper presents a formal mathematical analysis of Schaffer's contentions. This analysis confirms his major points, although it also shows that some of his experimental results must be interpreted with caution. Intuitively, those points of Schaffer's hold for a very simple reason: Unless you can *prove* from first principles what the correct priors are for a problem, you can prove essentially nothing concerning the likely efficacy of a particular learning algorithm, even if that learning algorithm involves complexity-penalization and/or cross-validation.[1]

Section 2 of this paper outlines the formalism which will be used in this analysis. Section 3 then investigates certain aspects of Schaffer's parity experiment. Section 4 then considers the ramifications of replacing Schaffer's error function with an off-training set error function, both for Schaffer's parity experiment and for his random-functions experiment. Finally, section 5 briefly discusses some extensions of this paper's analysis.

2. The extended Bayesian formalism

The formalism used in this paper is the extended Bayesian formalism introduced in (Wolpert, 1992) and used to analyze the "statistical mechanics" approach to learning in (Wolpert and Lapedes, 1992). This section describes the salient aspects of that formalism.

Label the input space as X, with elements x, and the output space as Y, with elements y. In the scenarios investigated by Schaffer, both X and Y are finite, with n and r elements, respectively. Label a training set of x-y pairs as L. $L_X(i)$ indicates the X component of the i'th pair in L, and similarly for $L_Y(i)$. The set of all the X values in L, all the $L_X(i)$, is written as L_X, and similarly for L_Y. There are m pairs in L altogether, m' of which are distinct.

Training sets are generated by sampling a target function f (taking $X \rightarrow Y$), usually with noise added. The entire process of generating L from the underlying function f is contained in the conditional probability distribution P(L | f). P(L | f) is called the "likelihood function" or the "sampling assumption", P(f) is called the "prior", and P(f | L) is called the "posterior".

After being "trained" on L, the learning algorithm outputs a hypothesis function h (taking $X \rightarrow Y$). h can often be viewed as the learning algorithm's guess for f, loosely speaking. If h goes through all the x-y pairs in L, h is said to "reproduce the training set". The relevant aspects of the learning algorithm are completely specified by the distribution P(h | L). (This is true regardless of how h is represented by the algorithm, be it in terms of trees, as in Schaffer's paper, or in some other manner.) For the algorithms considered in both this paper and Schaffer's, P(h | L) is deterministic, i.e., for a given L, P(h | L) = 0 for all but one h.

Since in general supervised learning is concerned with learning algorithms which only have access to L, P(h | L, f) = P(h | L) (see (Wolpert, 1992)). In other words, if f changes, yet L stays the same, the learning algorithm's behavior is unchanged. This is the only general restriction concerning P(f, h, L) explicitly invoked in this paper.

Conventional Bayesian analysis is the application of probability theory to the event space consisting of doubles (L, f). (Note that as used in such analysis, the term "hypothesis" actually refers to a target function f rather than an hypothesis function h.) Extended Bayesian analysis is the application of probability theory to the event space consisting of triples (L, f, h). In both frameworks, one usually wishes to inject an element of decision theory into the analysis. For current purposes, this can be done via a generalization *error function*, Er(f, h, L). This function measures the real-world "cost" associated with a triple (f, h, L). For example, if (as is often the case in noise-free scenarios) the cost is the average (over X) number of differences between f and h, then Er(f, h, L) = $\Sigma_x [\pi(x) \times (1 - \delta(f(x), h(x))]$, where $\pi(x)$ is the so-called "sampling distribution" and $\delta(., .)$ is the Kronecker delta function. In this paper I will be concerned with evaluating distributions of the form P(Er(., ., .) = E | something fixed.), since this is almost always the kind of distribution which is of interest in the real world, and since in particular it is the kind of distribution which Schaffer investigated.

As is shown in detail in (Wolpert, 1993), it is possible to express the frameworks of PAC (Valiant, 1984; Blumer et al., 1987; Blumer et al. 1987), Vapnik's uniform convergence work (Vapnik, 1982; Baum and Haussler, 1989), the "statistical mechanics" school (Tishby et al., 1989; Schwartz et al., 1990, van der Broeck, 1991), conventional Bayesian analysis (Berger 1985), etc., all in terms of the extended Bayesian formalism. (For example, both PAC and the statistical mechanics school concern themselves with evaluating properties of the distribution $P(E \mid f, m)$, given certain $P(L \mid f)$ and $P(h \mid L)$.) The reverse is not true; extended Bayesian analysis is more general than any of those other frameworks.

3. Schaffer's parity experiment

In general, one can not determine that any particular learning algorithm is to be preferred using only first principles reasoning. This was proven in (Wolpert, 1992), and the proof is simple enough to repeat here. First, taking L as fixed, view $P(h \mid L)$ as a vector of dimension r^n, with components indexed by the r^n possible hypothesis functions h. (Note that $P(h \mid L)$ is a list of r^n real numbers, one for each h.) View $P(f \mid L)$ similarly.

Theorem 1 *If Er(f, h, L) is invariant under the interchange of f and h, then P(E | L) can be written as a (non-Euclidean) inner product between the distributions P(h | L) and P(f | L).*

Proof: Write $P(E \mid L) = \Sigma_{h,f} [P(E \mid h, f, L) \times P(h, f \mid L)] = \Sigma_{h,f} [\delta(E, Er(h, f, L)) \times P(h, f \mid L)]$. This can be rewritten as $\Sigma_{h,f} [\delta(E, Er(h, f, L)) \times P(h \mid f, L) \times P(f \mid L)]$. However, as mentioned above, $P(h \mid f, L) = P(h \mid L)$; $P(E \mid L) = \Sigma_{h,f} [\delta(E, Er(h, f, L)) \times P(h \mid L) \times P(f \mid L)]$. Since by hypothesis $Er(f, h, L)$ is symmetric between f and h, the formula for $P(E \mid L)$ is just an inner product between the two vectors $P(h \mid L)$ and $P(f \mid L)$. QED.

Theorem 1 means that how well you generalize (i.e., the probability distribution over generalization error values) is determined by how "aligned" your learning algorithm $P(h \mid L)$ is with the actual posterior, $P(f \mid L)$. Unless one can somehow prove (!), from first principles, that $P(f \mid L)$ has a certain form, one can not "prove" that a particular $P(h \mid L)$ will be aligned with $P(f \mid L)$, and therefore one can not "prove" anything concerning how well that learning algorithm generalizes.[2] Note that if $Er(f, h, L)$ is not symmetric between f and h, $P(E \mid L)$ is no longer an inner product between $P(h \mid L)$ and $P(f \mid L)$. Nonetheless, even with such an error function, it is still true that how well you generalize is determined by how "aligned" $P(h \mid L)$ is with $P(f \mid L)$. (The only change is that "aligned" now has a different meaning.)

One particular instantiation of theorem 1 is the fact that one can not prove from first principles that a learning algorithm which tries to mitigate overfitting by introducing a preference for low-complexity h will work well. This, in essence, is the crux of Schaffer's conclusion based on his experiments. The implications of theorem 1 go further however. It shows that one can not prove, from first principles, that *any* desideratum results in better generalization, whether or not that desideratum concerns complexity and overfitting. So any argument claiming such a proof must be wrong. (See (Wolpert, 1992).)

It is hard to dispute the view that one is ultimately interested in $P(E \mid L)$, since what you know is L and what you want to know is E. The immediate implication is that considering anything else (e.g., $P(E \mid f, m)$) is misleading, as far as real-world generalization is concerned. (It is worth noting that conventional Bayesian analysis is concerned with $P(E \mid L)$, whereas other supervised learning formalisms like PAC and the uniform convergence formalism are not.) Certainly, at a minimum one should be careful in ascribing meaning to results based on any such non-$P(E \mid L)$ distribution. Nonetheless, and despite the fact that in many respects theorem 1 suffices to make his point, Schaffer doesn't directly concern himself with $P(E \mid L)$ in his experiments. Therefore we must go beyond the analysis of $P(E \mid L)$ presented in (Wolpert, 1992) to fully understand Schaffer's results.

The first experiment that Schaffer presents can be synopsized as follows: Let X be the vertices of a 5-dimensional hypercube (i.e., $X = \mathbf{B}^5$). Let Y be 0 or 1 (i.e., $Y = \mathbf{B}$). f is the parity function,

$y(x) = \Sigma_i x_i$ evaluated mod(2). m = 50 and is fixed throughout Schaffer's experiments, which means we can implicitly restrict things so that rather than probabilities over all triples {h, f, L}, we're only concerned with probabilities over those triples which have a 50-element L. Training sets are generated by noisy i.i.d. (independent identically distributed) sampling of f. The noise consists of replacing y(x) with ~y(x), randomly, with a certain probability z. More precisely, the likelihood, $P(L \mid f)$, can always be written as $P(L_Y \mid L_X, f) \times P(L_X \mid f)$. In Schaffer's experiments, $P(L_X \mid f)$ is given by i.i.d. sampling of X according to a sampling distribution $\pi(x \in X)$, i.e., $P(L_X \mid f) = \Pi^m_{i=1} \pi(L_X(i))$. $P(L_Y \mid L_X, f)$ reflects the noise, and is given by $\Pi^m_{i=1} P(L_Y(i) \mid L_X(i), f)$, where $P(L_Y(i) \mid L_X(i), f)$ equals z if $L_Y(i) = \sim f(L_X(i))$, and it equals (1 - z) if $L_Y(i) = f(L_X(i))$. As his error function Schaffer uses a variation of the error function discussed above. The modification is to allow the same noise which occurs in the creation of the training set to also arise in the testing set: $Er(f, h, L) = \Sigma_{x,y} [\pi(x) \times \{z + (1 - 2z)(\delta(y, f(x)))\} \times (1 - \delta(y, h(x)))]$.[3] The $\pi(x)$ in this error function is the same as the $\pi(x)$ which occurs in the likelihood.

Schaffer concerns himself with two learning algorithms. The first, which he calls the "naive" algorithm, tries to fit a decision tree directly to the training set, without any pruning. It is indicated by $P_n(h \mid L)$. The second, which he calls the "sophisticated" algorithm, starts with the same tree as the first, but then prunes, in an effort to avoid "overfitting". The degree of pruning is set by cross-validation. This algorithm is indicated by $P_s(h \mid L)$. Pruned trees are generically less "complex" than non-pruned ones, according to Schaffer's definition of complexity. Note that the parity function is a full tree, with 2^5 leaves, and therefore has the maximum complexity.

For both algorithms, Schaffer experimentally evaluates the average error, where the averaging is over training sets sampled from f. In other words, he evaluates $E(E \mid f, m) \equiv \Sigma_E [E \times P(E \mid f, m)]$ for the two algorithms. His conclusion is that, for the not too large z values he investigates, the naive algorithm has lower average error than the sophisticated algorithm. He interprets this as meaning that, for this target function at least, introducing a complexity cost to try to mitigate overfitting degrades performance. Since the sophisticated learning algorithm uses cross-validation to choose the complexity level, and is free to choose a large one, his results also indicate that in this scenario, cross-validation does not perform well.

These results of the first of Schaffer's experiments shouldn't be all that surprising. The parity problem is known to be a sticking point for many algorithms. For example, consider a learning algorithm which is local, in that its guess for what output goes with a question input value $q \in X$ only depends on the elements of L which lie close to q in X space. As an example, a K nearest neighbor algorithm with K fixed and small is local. For such an algorithm, it's often true that average error for test set inputs outside of the training set *increases* with training set size, for the parity target function, for low noise. This is also true for back-propagation, at least for small enough input spaces (see (Wolpert, 1991)). One shouldn't be too surprised to see parity rear its frustrating head in Schaffer's experiments as well.

However Schaffer's result would be expected to hold even if it weren't for the peculiar aspects of the parity target function. To see this, consider an algorithm even more "naive" than the one Schaffer used, in that it always produces maximally complex trees. This is the algorithm which always guesses h = the parity function, regardless of the data, $P(h \mid L) = \delta(h = \text{parity})$. Call this algorithm $P_p(h \mid L)$.

Theorem 2 *If $z \leq .5$, then independent of the sampling distribution, then independent of the sampling distribution, no learning algorithm has a lower $E(E \mid f, m)$ than an algorithm which always guesses maximally complex hypothesis functions, $P_p(h \mid L)$.*

Proof: First note that $P(E \mid f, L) = \Sigma_h [P(E \mid h, f, L) \times P(h \mid f, L)] = \Sigma_h [\delta(E, Er(f, h, L)) \times P(h \mid L)]$. Now if we temporarily restrict ourselves so that we only consider m-element training sets, we can write $P(E \mid f, m) = P(E \mid f) = \Sigma_L [P(E \mid f, L) \times P(L \mid f)]$. Combine our results to get $P(E \mid f, m)$ and then average E according to $P(E \mid f, m)$: $E(E \mid f, m) = \Sigma_{h,L} [Er(f, h, L) \times P(h \mid L) \times P(L \mid f)]$. For the algorithm $P_p(h \mid L)$, this equals $\Sigma_L [Er(f, h = f, L) \times P(L \mid f)]$. Since the error function is independent of L, this in turn equals Er(f, h = f). Now rewrite E(E | f, m) as $\Sigma_{h,L}$ [Er(f, h) $\times$ P(h, L | f)], which in turn equals Σ_h [Er(f, h) $\times$ P(h | f)], regardless of the learning algorithm. This result shows that for any learning algorithm, it is always true that E(E | f, m) $\geq \min_h$ Er(f, h). Since for the algorithm P_p(h | L) E(E | f, m) = Er(f, h = f), all we need to prove is that argmin_h Er(f, h) = f, under the conditions of the theorem. To do this we break up the extremization problem into n x-wise extremization problems. We fix x to x', and then we must show that for all x', the expression $\text{argmin}_{h(x')} \{ \Sigma_y [\{z + (1 - 2z)(\delta(y, f(x')))\} \times (1 - \delta(y, h(x')))]\} = f(x')$, under the conditions of the theorem. To do this, we need only consider the two possible values of h(x'). If h(x') = f(x'), the argument of the argmin equals z. If h(x') $\neq$ f(x'), that argument equals 1 - z. So long as z $\leq$ .5, the argument in the first case does not exceed the argument in the second case. QED.

It should be noted that nothing in the proof of theorem 2 depends on f being the parity function; all that's necessary is that the learning algorithm always guess the f specified in E(E | f, m). Note also that if z > .5, then no learning algorithm has a lower E(E | f = parity, m) than the algorithm $P_{\sim p}$(h | L), which always guesses the anti-parity function (y(x) = [1 + Σ_i x_i] evaluated mod(2)), regardless of the data. (The proof of this is essentially identical to the proof of theorem 2.) Like P_p(h | L), $P_{\sim p}$(h | L) also guesses maximally complex hypothesis functions. We have just proven the following:

Corollary 2.1 *If f is the parity function, then independent of the noise level and the sampling distribution, E(E | f, m) is minimized by a learning algorithm which guesses maximally complex hypothesis functions.*

If the noise is a function of x, z = z(x), we say that we have a "noise function" rather than a noise level. In such a scenario, theorem 2 still goes through, so long as $\max_x$ z(x) $\leq$ .5. Similarly, so long as $\min_x$ z(x) $\geq$ .5, no algorithm has a smaller value of E(E | f = parity, m) than $P_{\sim p}$(h | L).

So corollary 2.1 holds even if z is a function of x, provided either a) the noise is never less than .5 or b) the noise is never greater than .5. If the noise is "mixed", so that for some x $z(x) > .5$ while for other x $z(x) < .5$, then in general neither $P_p(h \mid L)$ nor $P_{\sim p}(h \mid L)$ is optimal. A simple example is where $z = 0$ for the set of $x \in X_0$, and $z = 1$ for other x. For this scenario, optimal guessing is to always guess a function that equals the parity function for those $x \in X_0$, but equals the anti-parity function for other x. This half-and-half function need not have maximal complexity, in general.

Schaffer doesn't compare $P_s(h \mid L)$ to $P_p(h \mid L)$; he compares it to $P_n(h \mid L)$. And although just like $P_p(h \mid L)$ $P_n(h \mid L)$ usually guesses h which are more complex than those guessed by $P_s(h \mid L)$, it doesn't always guess the parity function. However the underlying point Schaffer wishes to make - that algorithms which penalize complexity can do worse, for certain f - is proven formally in theorem 2. Theorem 2 also proves that any algorithm which so much as *allows* the possibility of not guessing the parity function must do worse than $P_p(h \mid L)$ (assuming $z(x) \le .5$ for all x). Given this, consider the technique of using cross-validation to choose amongst a fixed set of learning algorithms and then using the resultant learning algorithm. This technique, considered as a whole, is a mapping from training sets to hypothesis functions. In other words, it is a learning algorithm. Moreover, it is an algorithm which allows the possibility of not guessing the parity function (assuming some of the candidate learning algorithms don't guess the parity function). Accordingly, theorem 2 proves that when the target function is parity, using cross-validation will give worse results than just using $P_p(h \mid L)$, even if one of the candidate learning algorithms cross-validation considers is $P_p(h \mid L)$.

None of this should be too surprising. As Schaffer emphasizes however, there might be other target functions besides parity for which $P_n(h \mid L)$ (and perhaps even $P_p(h \mid L)$) is to be preferred, as compared either to cross-validation and/or to an algorithm which prefers low-complexity hypothesis functions. In some sense, the crucial question is how "likely" such target functions are in the real world.

4. Off-training set error

Note that theorem 2 implies that it is hard to beat a given learning algorithm, if that algorithm guesses an h which is assured of equalling the parity function for a large number of X values. Note also that one would expect $P_n(h \mid L)$ to guess an h which equals the parity function over most of the X values in L_X (all of those values, for z = 0). Accordingly, one might worry that since Schaffer used large training sets (as compared to the size of X), Schaffer's experiments are "biased" in favor of $P_n(h \mid L)$.

Schaffer responds to this, in a somewhat heuristic fashion. A more formal treatment, which touches on many of the points in Schaffer's response, can be had by using the approach presented in (Wolpert 1992). In that approach, one investigates the uniform P(f) case, to "filter out the biases" accompanying P(f)'s which contain peaks (Schaffer rediscovered this trick in his paper- see below). One also modifies the error function, turning it into an off-training-set error function: $Er_{off}(f, h, L) \equiv \Sigma_{y,x \notin L_X} [\pi(x) \times \{z + (1 - 2z)(\delta(y, f(x)))\} \times (1 - \delta(y, h(x)))] \;/\; \Sigma_{x \notin L_X} \pi(x)$. This error function does not give one credit for simply reproducing the training set. So by using it, one doesn't get the "bias" in favor of $P_n(h \mid L)$ referred to above. To view this another way, for zero noise, the error function Schaffer used has an upper bound which decreases with increasing m, for any learning algorithm which always reproduces its training set. $Er_{off}(., ., .)$ can be viewed as $Er(., ., .)$ "re-scaled" to remove this effect, and put all m on the same footing.

The original, pre-modification error function which Schaffer used in his parity experiments will be referred to as the "conventional" error function. From now on, unless stated otherwise, it will be assumed that the expression "$Er(., ., .)$" really means the off-training set error function $Er_{off}(., ., .)$, and not the conventional error function.

Replacing $Er(., ., .)$ with $Er_{off}(., ., .)$ has no effect on theorem 2.[4] So $P_p(h \mid L)$ is superior to other algorithms which prefer lower complexity h, even for off-training set error. However this modification of the error function does affect other things. In particular, it affects generalization behavior when one doesn't fix f (i.e., it affects generalization behavior when one admits the truth embodied in conventional Bayesian analysis - that in a real world experiment, you don't know f, and therefore must let if float). More precisely, with this error function, and with Schaffer's likelihood function, one has the following extension of a theorem from (Wolpert, 1992).

Theorem 3 *If P(f) is uniform, then regardless of the noise function and the sampling distribution, both $P(E \mid m)$ and $P(E \mid L)$ are independent of the learning algorithm.*

Proof: $P(E \mid m) = \Sigma_{h,f,L} [P(Er(f, h, L) = E \mid h, f, L, m) \times P(h, f, L \mid m)]$. This in turn can be written as $\Sigma_{h,f,L} [\delta(E, Er(h, f, L)) \times P(h, f, L)]$, where $\delta(., .)$ is the Kronecker delta function. Under the assumption that P(f) is uniform, this is proportional to $\Sigma_{h,f,L} [\delta(E, Er(h, f, L)) \times P(h \mid f, L) \times P(L \mid f)]$. This sum equals $\Sigma_{h,L} [P(h \mid L) \times \Sigma_f [\delta(E, Er(h, f, L))) \times P(L \mid f)]]$. Similarly, we can expand the distribution $P(E \mid L)$ as $\Sigma_h [P(h \mid L) \times \Sigma_f [\delta(E, Er(h, f, L)) \times P(f \mid L)]]$, which for uniform P(f) is proportional to $\Sigma_h [P(h \mid L) \times \Sigma_f [\delta(E, Er(h, f, L)) \times P(L \mid f)]]$. (In this case the constant of proportionality depends on P(L) and therefore depends on the likelihood, which along with P(f) is implicitly assumed to be held fixed as $P(h \mid L)$ varies.) I will show that the inner sum over f appearing in both the expression for $P(E \mid m)$ and that for $P(E \mid L)$ is a constant independent of h. This implies that $P(E \mid L)$ just equals that constant and is therefore independent of $P(h \mid L)$, as claimed (and similarly for $P(E \mid m)$). To prove that the inner sum is independent of h, expand it as the sum $\Sigma_f \{ \delta(\Sigma_{x \notin L_X, y} [\pi(x) \times \{z(x) + (1 - 2z(x))(\delta(y, f(x)))\} \times (1 - \delta(y, h(x)))], E \times \Sigma_{x \notin L_X} \pi(x)) \times P(L \mid f) \}$. For the likelihood Schaffer considers, $P(L \mid f)$ is independent of the values of f(x) for all $x \notin L_X$. Similarly, the value of the outer-most delta function is independent of the values of f(x) for all $x \in L_X$. Accordingly, write f as the union of f_1 and f_2, representing the values of f(x) for $x \in L_X$ and the values of f(x) for $x \notin L_X$, respectively. Doing this transforms our sum into a product of two sums. The first sum is Σ_{f_1} P(L | f_1), and is independent of h. The second sum is Σ_{f_2} { δ($\Sigma_{x \notin L_X, y}$ [$\pi(x) \times \{z(x) + (1 - 2z(x))(\delta(y, f_2(x)))\} \times (1 - \delta(y, h(x)))$], $E \times \Sigma_{x \notin L_X} \pi(x)$) }. To prove that this second sum is independent of h, first evaluate the inner sum over y, thereby getting Σ_{f_2} { δ($\Sigma_{x \notin L_X}$ [$\pi(x) \times \{z(x) + (1 - 2z(x))(\delta(\sim h(x), f_2(x)))\}$], $E \times \Sigma_{x \notin L_X} \pi(x)$) }. Now hold f_2 fixed, and consider the effect of replacing h with h^c, where $h^c(x') = \sim h(x')$ for one particular x' $\notin L_X$, while $h^c(x) = h(x)$ for all other $x \notin L_X$. By inspection, this results in a new value of the delta function which we could have gotten instead by leaving h unchanged and replacing $f_2(x)$ with ${f_2}^c(x)$, where ${f_2}^c(x') = \sim f_2(x')$ for x', while ${f_2}^c(x) = f_2(x)$ for all other $x \notin L_X$. In other words, δ($\Sigma_{x \notin L_X}$ [$\pi(x) \times \{z(x) + (1 - 2z(x))(\delta(\sim h^c(x), f_2(x)))\}$], $E \times \Sigma_{x \notin L_X} \pi(x)$) can be rewritten as δ($\Sigma_{x \notin L_X}$ [$\pi(x) \times \{z(x) + (1 - 2z(x))(\delta(\sim h(x), {f_2}^c(x)))\}$], $E \times \Sigma_{x \notin L_X} \pi(x)$). Now the mapping between f_2 and ${f_2}^c$ is one-to-one and onto over the set of all f_2's. Accordingly, performing this mapping before summing over all f_2's doesn't change the value of any such sum. This allows us to write our sum, Σ_{f_2} { δ($\Sigma_{x \notin L_X}$ [$\pi(x) \times \{z(x) + (1 - 2z(x))(\delta(\sim h(x), f_2(x)))\}$], $E \times \Sigma_{x \notin L_X} \pi(x)$) }, as Σ_{f_2} { δ($\Sigma_{x \notin L_X}$ [$\pi(x) \times \{z(x) + (1 - 2z(x))(\delta(\sim h(x), {f_2}^c(x)))\}$], $E \times \Sigma_{x \notin L_X} \pi(x)$) }. But we just finished proving that we can replace ${f_2}^c$ with f_2 if we also replace h with h^c; doing this gives Σ_{f_2} { δ($\Sigma_{x \notin L_X}$ [$\pi(x) \times \{z(x) + (1 - 2z(x))(\delta(\sim h^c(x), f_2(x)))\}$], $E \times \Sigma_{x \notin L_X} \pi(x)$) }. Therefore changing the value of h(x') for any x' $\notin L_X$ (i.e., replacing h with h^c) doesn't affect the value of our sum. Also, obviously, changing its value for any x' $\in L_X$ doesn't affect the value of this sum. If we now iterate this process of replacing h with h^c, only using different values of x', we see that changing h to any arbitrary function h' has no effect on our sum. This is what we needed to prove to establish the theorem. QED.

As an immediate consequence of theorem 3, neither $E(E \mid m)$ nor $E(E \mid L)$ varies with $P(h \mid L)$ if $P(f)$ is uniform. Both expectation values can be cast as an average over all f (using uniform $P(f)$), of the quantities $E(E \mid f, m)$ and $E(E \mid f, L)$, respectively. The picture that emerges then is that averaged over all f, all algorithms do equally well (as measured by either $E(E \mid f, m)$ or $E(E \mid f, L)$). So for any pair of learning algorithms, there are some f for which the first algorithm does better (at off-training set generalization), and some for which the second algorithm does better. (This is proven formally below.)

Now we don't know f beforehand, and accordingly we *must* average over f. Given that for uniform $P(f)$ such an average can't distinguish learning algorithms, we see that the *only* way that we can construct an algorithm which performs better than (the algorithm of) random guessing is if $P(f)$ is non-uniform, and our algorithm either explicitly or implicitly exploits that non-uniformity. The relative efficacy of different learning algorithms, be they anything from decision tree constructors to using cross-validation to choose amongst a set of learning algorithms, is completely determined by how accurately the algorithms encapsulate the non-uniformities of $P(f)$.

So theorem 3 provides another way of coming to one of the more important implications of theorem 1: unless you can rule out uniform $P(f)$, based on first-principles reasoning, you can not prove that one algorithm has a lower $E(E \mid m)$ or $E(E \mid L)$ than another, using only first-principles reasoning. This should be kept in mind whenever one encounters papers which appear to provide just such a first-principles proof in favor of a particular algorithm (e.g., (Baum and Haussler, 1989)).

Theorem 3 also formally justifies Schaffer's main point. Indeed, since that theorem has $P(f)$ uniform, it can be viewed as saying that the set of those f for which a generalizer preferring non-complex h beats one with no such preference is exactly offset by the set of f for which the reverse is true (regardless of the complexity measure used).

Schaffer was aware of the utility of one of the quantities addressed by theorem 3, $E(E \mid m)$ with uniform $P(f)$. In fact, in his second set of experiments, rather than fixing f, Schaffer repeatedly chose the target function at random (according to a uniform $P(f)$), from the set of all 2^{2^5} possible such functions. He then compared the average behavior of the naive algorithm to that of the sophisticated one, averaged over all such f and over training sets generated from f. In other words, he measured E(E | m) with a uniform P(f). (For a discussion of what P(f) might "mean in the real world", as opposed to in experiments like Schaffer's, see appendix E in (Wolpert, 1992).)

Unfortunately though, although theorem 3 makes Schaffer's point, it also partially negates the significance of his second set of experiments. Just as he did in his parity experiment, for the second set of experiments Schaffer used the conventional error function rather than the off-training-set error function. His result was that the naive algorithm did better than the sophisticated one. However theorem 3 proves that the two algorithms have to perform exactly equally for off-training set error. Consequently, the result Schaffer saw must be due to the fact that the naive algorithm performs better for the elements inside the training set.

Given this, Schaffer's result doesn't have much to say about the merits of pruning decision trees. This is made most clear by considering the z = 0 case; when there's no noise, the only way pruning could *possibly* be of assistance is if it results in improved off-training set guessing, since it's assured of giving worse on-training set guessing. This fact has nothing to do with theorem 3, and comes as no surprise to even the most die-hard advocate of pruning. Given this fact, if one sets up a scenario in which off-training set behavior is "filtered out", so that only on-training set behavior can distinguish the various learning algorithms, one has fixed things so that pruning schemes must perform poorly. Theorem 3 shows that Schaffer's second experiment was just such a scenario.

Intuitively, theorem 3 holds because f is not fixed in the distributions theorem 3 considers. With uniform P(f), and f not fixed, one off-training-set pattern for f is just as likely as any other, regardless of L. Accordingly, in the scenarios considered by theorem 3, any "biases" in one's learning algorithm, towards guessing certain off-training set patterns for f, can't possibly help as far as off-training set error is concerned. Now this general property will not hold if one fixes f in the distribution of interest. For example, certain learning algorithms perform better than others if the distribution of interest is P(E | f, m) (see theorem 2). Interestingly though, theorem 3 can be used to set limits on such distributions which have f fixed, even though theorem 3 has f completely free to float:

Theorem 4 *For any two learning algorithms $P_1(h \mid L)$ and $P_2(h \mid L)$, independent of the noise function and sampling distribution,*

i) if there exists an f such that $E(E \mid f, m)$ is lower for $P_1(h \mid L)$, then there exists a different f such that $E(E \mid f, m)$ is lower for $P_2(h \mid L)$;

ii) if there exists an f and an L such that $E(E \mid f, L)$ is lower for $P_1(h \mid L)$, then there exists a different f and L such that $E(E \mid f, L)$ is lower for $P_2(h \mid L)$;

iii) if there exists a P(f) and an L such that $E(E \mid L)$ is lower for $P_1(h \mid L)$, then there exists a different P(f) and L such that $E(E \mid L)$ is lower for $P_2(h \mid L)$;

iv) if there exists a P(f) such that $E(E \mid m)$ is lower for $P_1(h \mid L)$, then there exists a different P(f) such that $E(E \mid m)$ is lower for $P_2(h \mid L)$

Proof: Examine $\Sigma_f E(E \mid f, m)$. This sum can be rewritten as $r^n \times \Sigma_f [E(E \mid f, m) \times P(f \mid m)]$, if we take a uniform $P(f \mid m)$. In turn, write $\Sigma_f [E(E \mid f, m) \times P(f \mid m)] = \Sigma_{f,E} [E \times P(E \mid f, m) \times P(f \mid m)] = E(E \mid m)$. Therefore we can evaluate $\Sigma_f E(E \mid f, m)$ by evaluating $E(E \mid m)$ for a uniform prior. By theorem 3, such a $E(E \mid m)$ is independent of $P(h \mid L)$. Therefore we have shown that the sum we're interested in, $\Sigma_f E(E \mid f, m)$, is independent of $P(h \mid L)$. Accordingly, if for one f $E(E \mid f, m)$ is lower for $P_1(h \mid L)$ than it is for $P_2(h \mid L)$, then there must be another f for which the reverse is true. This proves (i). To prove (ii), indicate the value of $E(E \mid f, L)$ occurring with learning algorithm $P_1(h \mid L)$ as $E_1(E \mid f, L)$, and similarly for $P_2(h \mid L)$. By hypothesis, there exists an f and L, f* and L*, such that $E_1(E \mid f^*, L^*) < E_2(E \mid f^*, L^*)$. Now write $E(E \mid f, m) = \Sigma_L [E(E \mid f, L) \times P(L \mid f)]$, which means that $E_1(E \mid f^*, m) - E_2(E \mid f^*, m) = \Sigma_L [\{E_1(E \mid f^*, L) - E_2(E \mid f^*, L)\} \times P(L \mid f^*)]$. If $E_1(E \mid f^*, m) \geq E_2(E \mid f^*, m)$, then since $E_1(E \mid f^*, L^*) < E_2(E \mid f^*, L^*)$ there must an $L \neq L^*$ such that $E_1(E \mid f^*, L) > E_2(E \mid f^*, L)$, and we're done. If instead $E_1(E \mid f^*, m) < E_2(E \mid f^*, m)$, then by (i) we know that there must be a different f, f ', for which $E_1(E \mid f', m) > E_2(E \mid f', m)$. By our expansion for $E(E | f, m)$, we know that for this to be true there must be an L such that $E_1(E | f', L) > E_2(E | f', L)$. This proves (ii). To prove (iii), write $E_i(E | L) \equiv \Sigma_f [E_i(E | L, f) \times P(f | L)]$, so that $E_2(E | L) - E_1(E | L) = \Sigma_f [\{E_2(E | L, f) - E_1(E | L, f)\} \times P(f | L)]$. Since this is > 0 by hypothesis, there exists an L* and an f* such that $E_2(E | L^*, f^*) - E_1(E | L^*, f^*) > 0$. By (ii), this means that there exists a different L' and f ' such that $E_2(E | L', f') - E_1(E | L', f') < 0$. If we now set $P(f) = \delta(f - f')$, we get $E_2(E | L') - E_1(E | L') = \{E_2(E | L', f') - E_1(E | L', f')\} \times P(L' | f') / P(L')]$, which is < 0. This proves (iii). To prove (iv), write $E_i(E | m) \equiv \Sigma_f [E_i(E | f, m) \times P(f | m)]$. This means that we can write $E_2(E | m) - E_1(E | m) = \Sigma_f [\{E_2(E | f, m) - E_1(E | f, m)\} \times P(f | m)]$. Since this is > 0 by hypothesis, there exists an f* such that $E_2(E | f^*, m) > E_1(E | f^*, m)$. By (i), this means that there exists an f ' such that $E_1(E | f', m) > E_2(E | f', m)$. If we now set $P(f) = \delta(f - f')$, and plug into our expansion for $E(E | m)$, we have $E_1(E | m) > E_2(E | m)$. QED.

Theorem 3 and its implications are not too different from the heuristic arguments Schaffer presents in his section 5.1. Perhaps the main advantage of theorem 3 over those arguments is that, being completely formal, one can exploit theorem 3 to generate other formal results, like those making up theorem 4.

Note that no restrictions whatsoever concerning $P_1(h | L)$ and $P_2(h | L)$ are assumed in theorem 4. Accordingly, theorem 4 means that no algorithm can be assured of always doing worse (off the training set) than another one. Phrased differently, it means that for any two learning algorithms, there’s always a scenario in which the first algorithm beats the second one.

However this does not mean that for any learning algorithm, there’s always a scenario in which that algorithm beats all other algorithms *simultaneously*. For example, it is not true that for any $P(h | L)$, there exists an f such that $E(E | f, m)$ is minimized for that $P(h | L)$. To see this, note that theorem 2 uses no property of the parity function, i.e., it is always true that the algorithm $P(h | L) = \delta(h = f^*)$ for some particular f* will minimize $E(E | f = f^*, m)$. Now consider a “spread” learning algorithm for which $P(h | f)$ always extends over more than one h (i.e., for which for any particular f, $P(h \mid f) \neq 0$ for more than one h.) For such a learning algorithm, for $z < .5$, theorem 2 becomes a strict inequality, i.e., $E(E \mid f^*, m)$ for the algorithm $P(h \mid L) = \delta(h = f^*)$ is always lower than it is for a spread learning algorithm. According, one can not find an f such that a spread learning algorithm will minimize $E(E \mid f, m)$ for any f; the learning algorithm $P(h \mid L) = \delta(h = f)$ will always do better than that spread learning algorithm, for that f.

So all learning algorithms are the same in that: 1) all algorithms have the same average error, when the average is uniform over all f; 2) no learning algorithm can beat another one for all f. However learning algorithms can differ in that: 1) for non-uniform P(f), different algorithms have different average error; 2) for some algorithms there is an f for which that algorithm is optimal (i.e., for which that algorithm beats all other algorithms), but some other algorithms are not optimal for any f.[5]

5. Variations

Generically, in the limit of $n >> m$, the probability that a random X value from a testing set lies within L_X goes to 0, which usually implies that off-training-set error becomes equivalent to conventional error in that limit. For such scenarios, theorems 3 and 4 imply that no algorithm is generally superior to any other one, as measured by $P(E \mid f, m)$, even if one uses the conventional error function. This is particularly interesting in light of the fact that PAC and the statistical mechanics formalism both concern themselves with $P(E \mid f, m)$, and both use the conventional error function. More generally, guided by the insight provided by theorems 1, 3 and 4, one can return to the many papers which appear to claim first-principles justification for a particular learning algorithm, and scrutinize them for the seemingly innocuous assumption(s) which (according to theorems 1, 3 and 4) they must make to get their results. (Unfortunately, although this process can shed light on the formalisms used in such papers, it tends to be quite laborious.)

Recall that the technique of using cross-validation to choose amongst a fixed set of learning algorithms is itself a learning algorithm. The immediate implication is that everything said so far in this paper applies just as well to the use of cross-validation as it does to learning algorithms which favor non-complex h. Accordingly, cross-validation can not be justified using only first principles reasoning, and one can always find scenarios in which it fails.

However one does not need to view cross-validation as a learning algorithm to see this. It is possible to modify the formalism presented in this paper so that rather having probabilities over triples {f, h, L}, one has probabilities over triples {f, g, L}, where g is the learning algorithm one chooses, that choice being determined by L via $P(g \mid L)$. (As opposed to h being the hypothesis function one chooses, that choice being determined by L via $P(h \mid L)$.) The technique of cross-validation is one particular $P(g \mid L)$, just as a g is one particular $P(h \mid L)$. One can perform the analysis using {f, g, L} rather than the space exploited in this paper, {f, h, L}. One gets essentially the same results, except that h gets replaced with g. This reflects the fact that there is a close formal parallel between the analysis of learning based on probabilities over {f, g, L} and that based on probabilities over {f, h, L}. (For a discussion of the many implications of this formal parallel, see the section on meta-generalization in (Wolpert, 1992).)

These results concerning cross-validation might seem peculiar, since there are a number of quite reasonable heuristic “justifications” for cross-validation. For example, consider a scenario in which we have k learning algorithms, $G_1, ..., G_k$, and a target function $f = f^*$. Suppose that it’s clear that for most training sets of size m or slightly smaller constructed by sampling f^*, G_1 performs a good deal better at off-training-set guessing than do the other k - 1 algorithms. For such a scenario, one would expect cross-validation to work (i.e., to optimize generalization), if one averages over training set chosen from f^*. This is because cross-validation will choose G_1, on average, and G_1 is also the best of the k algorithms to use with the entire training set. Now one might expect this kind of situation to be generic, in which case cross-validation would work, generically. What the theorems of this paper prove however is that this situation is *not* generic. For fixed $G_1, ..., G_k$, although cross-validation works for $f = f^*$, there must be other f for which it fails. This is true whether one judges “failure” by examining behavior with a particular training set or if one instead averages over training sets.

As a final comment, it should be noted that there are, of course, many other error functions besides the ones considered in this paper. There are also many other ways that one can use the quantity P(E | stuff) to measure the efficacy of a particular learning algorithm, besides using it as in this paper, to get an expectation value. (E.g., one might wish to examine P(E = 0 | stuff) rather than Σ_E [E × P(E | stuff)].) Such alternatives were not explored in Schaffer's paper however, and they also have not been addressed in a systematic way in the machine learning community. This is why they are beyond the scope of this paper.

Acknowledgments

This work was supported in part by NLM grant F37 LM00011-01. I would like to thank Wray Buntine for helpful comments. I would also like to thank Cullen Schaffer for going over this manuscript.

Notes

1. No matter how reasonable one's arguments for a particular set of priors, the simple fact of the matter is that those priors might be wrong. (Somewhat disingenuously, some conventional Bayesians refer to such a situation where one's priors are wrong as "an opportunity to learn".) That is why the requirement is to *prove* certain prior probabilities. This does not mean *argue* for certain prior probabilities, as properly reflecting our "prior beliefs". The distinction is made explicit by the mathematics presented in this paper. In particular, by the simple expedient of treating them as distinct objects, that mathematics does not allow us to confuse our prior beliefs concerning a quantity with the prior probability over that quantity, a mistake commonly made in conventional Bayesian analysis. See section 2 below.

2. One must be careful not to misinterpret $P(f \mid L)$. In particular, one should not take the view that it is just a "degree of belief" in f, and therefore can be set to whatever I (the believer) wish. In this formalism, personal beliefs, concerning how probable a function is (or anything else), are embodied in $P(h \mid L)$, which need not have anything to do with $P(f \mid L)$. $P(f \mid L)$ should instead be thought of, intuitively, as the "true" probability of f given L. (The contrast is much like that between the "true" input-output function f and one's guessed function, h.) More formally, to interpret $P(f \mid L)$ as well as $P(h \mid L)$ in terms of one's personal degree of belief amounts to an extra assumption. This assumption is a restriction on the probability distribution across (f, h, L), to reflect the fact that $P(f \mid L)$ and $P(h \mid L)$ are so intimately related. No such extra assumption concerning $P(f, h, L)$ is needed in this paper. Indeed, such an assumption would be a case of one's subjective statistical preferences and philosophy (concerning what $P(f \mid L)$ "means") being imposed on the mathematics, rather than letting the mathematics speak for itself. Such imposing-of-philosophy is the very sin for which conventional Bayesians so vehemently chastise sampling theory statisticians. See appendix E in (Wolpert, 1992).

3. For those terms in the sum for which $y = f(x)$, the expression inside the { } equals 1 - z, and for those terms for which $y \neq f(x)$, it equals z. So that expression gives the probability that a given pair in the test set has output value y, given that it was produced by sampling f(x), with noise z.

4. As in the proof of theorem 2, we have $E(E \mid f, m) = \Sigma_{h,L} [Er(f, h, L) \times P(h \mid L) \times P(L \mid f)]$, which for the algorithm $P_p(h \mid L)$ equals $\Sigma_L[Er(f, h = f, L) \times P(L \mid f)]$ for f the parity function. On the other hand, we can rewrite $E(E \mid f, m)$ as $\Sigma_L[P(L \mid f) \times \Sigma_h [Er(f, h, L) \times P(h \mid L)]]$, which implies that $E(E \mid f, m) \geq \Sigma_L [P(L \mid f) \times \min_h [Er(f, h, L)]]$. As in the proof of theorem 2, by breaking it up into (n - m') x-wise minimization problems, we see that for $z(x) \leq .5$, $\min_h [Er(f, h, L)] = Er(f, h = f, L)$ (where for the moment we're implicitly only concerning ourselves with $h(x \notin L_X)$, since the values

of $h(x \in L_X)$ have no effect on the value of Er(f, h, L)). Accordingly, E(E | f, m) is bounded below by Σ_L [P(L | f) × Er(f, h = f, L)], which as mentioned above is just the value of E(E | f, m) when the learning algorithm is $P_p(h \mid L)$. QED.

5. There exist a number of interesting variations of this issue of whether there exists a scenario in which a particular learning algorithm can be optimal. For example, one might wish to restrict the discussion to certain kinds of algorithms (e.g., algorithms which are not “spread”), or one might define “optimal” differently (e.g., as whether there exists a P(f) such that E(E | m) is minimized), etc. All such variations are beyond the scope of this paper.

References

Baum, E., and Haussler, D. (1989). What size neural net gives valid generalization? *Neural Computation*, **1**, 151-160.

Berger, J. (1985). Statistical decision theory and Bayesian analysis. Springer-Verlag.

Blumer, A., et alia (1987). Occam’s razor. *Information Processing Letters*, **24**, 377-380.

Blumer, A., et alia (1989). Learnability and Vapnik-Chervonenkis dimension. *Journal of the ACM*, **36**, 929-965.

Buntine, W. (1992). Learning classification trees. *Artificial intelligence and statistics (Proceedings of conference on AI and statistics, Florida, 1991)*. D. Hand (Ed.)

Efron, B. (1979). Computers and the theory of statistics: thinking the unthinkable, *SIAM REVIEW*, **21**, 460-480.

Schaffer, C., (1992a). Sparse Data and the Effect of Overfitting Avoidance in Decision Tree Induction, *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, 1992.

Schaffer, C., (1992b). Deconstructing the Digit Recognition Problem. *Machine Learning: Proceedings of the Ninth International Conference (ML92)*, D. Sleeman and P. Edwards (Eds), Morgan Kaufmann, 1992.

Schaffer, C., (1993). Overfitting avoidance as bias. *Machine Learning*, **10**, 153-178.

Schwartz et alia (1990). Exhaustive Learning. *Neural Computation*, **2**, 374-385.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions, *J. Roy. Statistic. Soc. B*, **36**, 111-120.

Stone, M. (1977a). An asymptotic equivalence of choice of model by cross-validation and Akaike's criterion. *J. Roy. Statistic. Soc. B*, **39**, 44-47.

Stone, M. (1977b). Asymptotics for and against cross-validation, *Biometrika*, **64**, 29-35.

Tishby, N., Levin, E., and Solla, S. (1989). Consistent inference of probabilities in layered networks: predictions and generalization. *Proceedings of the international Joint Conference on Neural Networks*, Washington, D.C., pp. II 403-409, IEEE.

Valiant, L. (1984). A theory of the learnable. *Communications of the ACM*, **27**, 1134-1142.

Van der Broeck, C., and Kawai, R. (1991). Generalization in feedforward neural and Boolean networks. Presented at *AMSE International conference on neural networks*, San Diego, CA.
Vapnik, V. (1982). Estimation of dependences based on empirical data. Springer-Verlag.

Wolpert, D. (1991). Improving the performance of generalizers via time-series-like pre-processing of the learning set. Los Alamos Laboratory Report LA-UR-91-350.

Wolpert, D. (1992). On the connection between in-sample testing and generalization error. *Complex Systems,* **6**, 47-94.

Wolpert D., and Lapedes, A. (1992). An investigation of exhaustive learning. Santa Fe Institute report 92-04-20. Submitted.

Wolpert, D. (1993). The relationship between the various supervised learning formalisms. To appear in *The future of supervised learning.* D. Wolpert (Ed.). Addison-Wesley.