From: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence,
(895-900), AAAI Press, 1993

Estimating the Accuracy of Learned Concepts*

Timothy L. Bailey and Charles Elkan
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, California 92093-0114

Abstract

This paper investigates alternative estimators of the accuracy of concepts learned from examples. In particular, the cross-validation and 632 bootstrap estimators are studied, using synthetic training data and the FOIL learning algorithm. Our experimental results contradict previous papers in statistics, which advocate the 632 bootstrap method as superior to cross-validation. Nevertheless, our results also suggest that conclusions based on cross-validation in previous machine learning papers are unreliable. Specifically, our observations are that (i) the true error of the concept learned by FOIL from independently drawn sets of examples of the same concept varies widely, (ii) the estimate of true error provided by cross-validation has high variability but is approximately unbiased, and (iii) the 632 bootstrap estimator has lower variability than cross-validation, but is systematically biased.

1 Introduction

The problem of concept induction (also known as the classification problem [Kononenko and Bratko, 1991] and known as the prediction problem in the statistical literature [Efron, 1983]) is perhaps the most intensively studied topic in machine learning. Given a training set of classified examples drawn from a certain domain, and a language for stating concept definitions, the problem is to invent a concept definition inspired by the training set that will correctly classify new examples drawn from the domain.

Various methods have been proposed to evaluate the accuracy of learned concept definitions. Most researchers have used methods whose objective is, explicitly or implicitly, to approximate what we call true error: the expected error of the learned concept on new examples. Cross-validation is the method most often used, as for example in [Towell *et al.*, 1990] and [Weinstein *et al.*, 1992]. However several Monte Carlo studies of cross-validation and alternative methods have been done [Fitzmaurice *et al.*, 1991; Sanchez and Cepeda, 1989; Efron, 1983], which tend to indicate that cross-validation is inferior to the other methods.

Compared to previous studies of methods for evaluating the performance of learning algorithms, this paper has several novelties. First, in addition to investigating the variance and bias of estimators of true error, we also examine how randomness in the choice of training examples leads to variance in true error itself. To do this we need to be able to evaluate true error exactly, which we achieve with synthetic data sets of examples of known concepts. Knowledge of exact true error rates also underlies the second novelty of our work, which is that we investigate the correlation between estimators of error and true error, distinguishing between downward (optimistic) and upward (pessimistic) bias. The third novelty here is the learning algorithm used, FOIL[Quinlan, 1990]. Logical rule learning, sometimes called inductive logic programming, is an active research area currently, and FOIL and its variants are the most widely used rule-learning algorithms. ¿From a more general point of view, the novelty of FOIL is that it is almost completely insensitive to the presence of duplicate examples in a training set, unlike the learning algorithms used in previous studies.

The rest of this paper is laid out as follows. The "true error" metric for evaluating learned concepts and methods of estimating this metric are described in Section 2. The FOIL learning algorithm and the synthetic data sets used in our experiments are the topic of Section 3. Our experimental results are presented in Section 4, and finally, our conclusions appear in Section 5.

2 Measuring learned concept accuracy

In the concept induction problem, the learning algorithm is given as input a training set X consisting of a set of examples $x_1, x_2, \ldots, x_n$ where each example consists of two parts $x_i = (t_i, y_i)$, where t_i is a vector of attributes and y_i is the class. The algorithm constructs a concept definition that uses the attributes to predict the class. We assume that the training set is selected randomly ac-

*This work was supported in part by the National Science Foundation under Award No. IRI-9110813. Timothy Bailey is supported by an NIH Human Genome Project predoctoral training grant.

cording to some (unknown) probability distribution F.[1]

For the classification problem, the most commonly used criterion for the goodness of a concept definition is its true error rate: the probability that the concept definition will incorrectly classify an example drawn randomly from the same distribution F as the training set. This is sometimes referred to as the *generalization error* of the concept definition since it measures how well the concept definition generalizes to examples the learning algorithm did not see.

The definition of true error. The following definitions are essentially the same as those in [Efron, 1983]. Suppose a learning algorithm constructs prediction rule $\eta(t, X)$ from training set X. Let $\eta_i = \eta(t_i, X)$ be the prediction of this rule on example x_i and let $Q[y_i, \eta_i]$ be the error of the learned rule on that example. Formally

$$\begin{aligned} Q[y_i, \eta_i] &= 0 \text{ if } \eta_i = y_i \\ &= 1 \text{ if } \eta_i \neq y_i. \end{aligned}$$

The true error rate Err is then the probability of incorrectly classifying a randomly selected example $X_0 = (T_0, Y_0)$, which is the expectation

$$Err = E_F\ Q[Y_0, \eta(T_0, X)].$$

Readers familiar with PAC learning theory will notice that this definition of true error rate subsumes the definition of the error of a hypothesis with respect to a target concept given in [Haussler, 1988] and often used by PAC theorists. The difference is that the PAC framework usually assumes that the examples are either "in" or "out" of a hypothesis: that the hypothesis and target concept are deterministic. We do not make that assumption about either the learned concept or the target concept (i.e. the source of examples). It is quite possible that two examples may have the same attribute values and yet have different classes in our framework.

The issue of estimating true error. We must usually estimate true error rate Err since the distribution F is usually unknown. The most conceptually straightforward way to do this is to randomly draw a test set of examples (independent of the training set) and take the mean error to be the test error

$$Terr = \frac{1}{m} \sum_{i=1}^{n} Q[y_i, \eta(t_i, X)]$$

where m is the size of the test set. $Terr$ approaches Err as $m \to \infty$. Unfortunately, this test set method of estimating Err is often infeasible because collecting examples is expensive. Also, if the number of examples available is limited, the user of the learning algorithm will want to use them all in constructing the classification rule. Of course, if data is cheap and Err is very low, the test set approach may be used.

Several methods for estimating Err exist for use when the test set method is unattractive. These methods use the training set X itself as the basis for estimating Err. [Efron, 1979] shows that the three methods known as bootstrap, jackknife and cross-validation are mathematically related. Bootstrap is a nonparametric maximum likelihood estimator,[2] jackknife is a quadratic approximation to bootstrap, and cross-validation is similar in form and value to jackknife. Later work [Efron, 1983] argues empirically and analytically that a modified bootstrap method known as 632 bootstrap is superior. Therefore, we focused on cross-validation and 632 bootstrap estimators in our work.

The cross-validation method. Cross-validation estimates Err by reserving part of the training set for testing the learned theory. In general, v-fold cross-validation (randomly) splits the training set into v equal-sized subsets, trains on $v-1$ subsets and tests on one subset. Each subset is left out of the training set and used as the test set once. A common choice for v is the size n of the original training set. Since each subset is then a singleton, this is called "leave-one-out" or n-fold cross-validation.

For a given amount of training data, leave-one-out cross-validation allows learning from the largest possible number of examples, while still basing the estimation of accuracy on unseen data. Intuitively, the true error of concepts learned on $n-1$ examples during leave-one-out cross-validation should be close to what we are trying to estimate, which is the true error of the concept learned from all n examples. Other methods which use smaller subsets for training, in particular v-fold cross-validation where $v < n$, should intuitively be poorer estimates of Err when the number of training examples available is small.

The 632 bootstrap method. The 632 bootstrap technique [Efron, 1983] for estimating Err creates a "resample" from a training set by choosing n samples *with replacement* from the training set. Resamples are typically multisets. The 632 bootstrap estimator is defined as $e_{632} = 0.368 e_r + 0.632 e_b$ where e_b is the proportion of the examples not chosen in the resample that are misclassified by the rule learned on the resample, and e_r is the proportion of training set examples which are misclassified by the rule learned on the whole training set. In practice, e_b is averaged over many resamples.

Previous comparisons of methods. The 632 bootstrap method is reported in [Efron, 1983] to estimate true error better in five experiments than several other methods including the original bootstrap method and cross-validation. More recent and comprehensive experiments using linear discriminant classifiers confirm the good performance of the 632 bootstrap method [Fitzmaurice *et al.*, 1991; Sanchez and Cepeda, 1989]. The main criterion for evaluating an estimator $\hat{Err}$ in these papers is mean squared error, defined as MSE = E$(\hat{Err} - Err)^2$. The MSE of an estimator is a combination of its bias and variance, and is insensitive to whether bias is upward or downward. Here we examine variance, bias, and the direction of bias separately.

[1] Notice that the distribution of the examples, F, is over both the class and the attributes of the examples. In much work in machine learning the distribution is just over the attributes, and the class is assumed to be deterministically dependent on the attributes. Our definition subsumes the usual definition as a special case.

[2] The bootstrap estimate of Err replaces the true distribution F with its nonparametric maximum likelihood estimate $\hat{F}$ where $\hat{F}$ is the empirical probability distribution putting equal mass $1/n$ on each observed sample x_i.

In experiments using nearest neighbor classifiers, both cross-validation and the 632 bootstrap method have been reported to perform poorly, and a composite method has been suggested [Weiss, 1991; Weiss and Kulikowski, 1991]. Linear classifiers and nearest neighbor methods are very different from symbolic concept induction methods such as FOIL or decision tree algorithms. In almost all symbolic learning work cross-validation has been used to estimate the accuracy of learned concepts. One exception is work using the CART decision tree learning algorithm, for which the original and 632 bootstrap methods are compared with cross-validation in [Crawford, 1989], with the conclusion that 632 bootstrap is best.

3 Experimental framework

This section describes the learning algorithm and the data sets that we used to study the performance of cross-validation and the 632 bootstrap method as estimators of the true error of learned concepts.

The FOIL algorithm. This algorithm [Quinlan, 1990] produces concept definitions which are sets of function-free Datalog-with-negation clauses. Training sets given to FOIL are encoded as relation extensions, i.e. as lists of ground tuples. One or more of the relations is designated as the target relation, and FOIL attempts to learn an intensional definition for it in terms of the other relations.

For example, FOIL might be given the relations *linked-to(X,Y)* and *can-get-to(X,Y)* defined extensionally and asked to find an intensional definition of *can-get-to(X,Y)*. If the extensions of the two relations are such that *can-get-to(X,Y)* is the transitive closure of *linked-to(X,Y)*, FOIL may succeed in finding the intensional definition,

$$\begin{array}{lcl} \textit{can-get-to}(X,Y) & \leftarrow & \textit{linked-to}(X,Y) \\ \textit{can-get-to}(X,Y) & \leftarrow & \textit{linked-to}(X,Z), \textit{linked-to}(Z,Y). \end{array}$$

There are many possible encodings for a given classification problem. The particular one chosen can greatly affect the efficiency of FOIL and whether or not a concept definition is found at all. One advantage of FOIL over other learning algorithms is that background knowledge can be provided in the form of additional relations that can be used in forming the concept definition.

FOIL uses a greedy algorithm that builds concept definitions one clause at a time. Each clause is built one literal at a time, trying all possible variabilizations of each possible relation, and adding the one with the highest "information gain" to the clause. There is limited within-clause backtracking: if no literal can be found with positive information gain, the algorithm removes the last literal added to the clause and replaces it with another candidate with positive (but lower) gain.

Synthetic data sets. We constructed synthetic data sets in order to be able to evaluate true error exactly. These data sets were designed to be similar to real molecular biology data sets that we extracted from the EPD eukaryotic promoter genetic sequence database for other work. Each synthetic data set contained positive and negative examples of a single, short, disjunctive normal form concept (DNF). Each example was defined by 50 binary attributes. The DNF concepts chosen were all short (i.e., they contained few clauses and the clauses were short), so the majority of the 50 attributes are uncorrelated with the concept: they are "noise" or "irrelevant" attributes. The actual concepts used are as follows.

name	*DNF formula*
dnf1	$(B \wedge C) \vee (\overline{A} \wedge C \wedge D) \vee (\overline{C} \wedge B) \vee (A \wedge \overline{D})$
dnf2	$(B \wedge C \wedge \overline{D} \wedge E) \vee (\overline{A} \wedge F \wedge G \wedge \overline{H})$
dnf3	$B \wedge C \wedge \overline{D} \wedge \overline{E}$
dnf4	$(\overline{B} \wedge C) \vee (\overline{A} \wedge C \wedge D)$

The distribution of training examples for each concept was the same. We randomly selected examples from a mixture distribution with positive and negative examples having equal probability. In other words, to generate an example we randomly chose to generate either a positive or negative example (with equal probability) and then randomly generated binary strings of length 50 until an example of the correct class was found. This example was then put in the training set and the process was repeated until the desired number of examples had been generated.

4 Experimental results

We ran a number of Monte Carlo simulations using synthetic data sets and FOIL as just described. In each experiment we measured true error rate Err, its cross-validation estimate, and its 632 bootstrap estimate. In brief, we discovered that 632 bootstrap performs very poorly with this learning algorithm and these target concepts. 632 bootstrap had lower variance than cross-validation, but it had very poor correlation with Err and was strongly biased.

Experimental design. We performed a number of experiments using data generated by the target concepts dnf1, dnf2, dnf3 and dnf4. The basic procedure in each experiment for each distribution was, using FOIL as the learning algorithm, as follows. Except in Figure 1, because of space limitations results are plotted below for dnf1 only. Qualitatively similar results were always obtained for each concept.

1. Generate a training set from the distribution.
2. Learn one concept from the *entire* training set.
3. Compute true error Err of this learned concept.
4. Estimate Err by leave-one-out cross-validation.
5. Estimate Err by the 632 bootstrap method.

Computing the value of Err was accomplished by exhaustively comparing the value of the learned concept definition and the true concept definition for every possible combination of values of the *relevant* attributes. The relevant attributes were those mentioned in the true concept definition or in the learned concept definition. In general, the number of relevant attributes was only about 10, making it possible to exhaustively check all 2^{10} possible combinations. It would have been prohibitive to check all 2^{50} possible combinations of attributes.

Scatter plot experiments. In these experiments, the basic procedure was repeated 100 times for each distribution. The size of the training set in each experiment

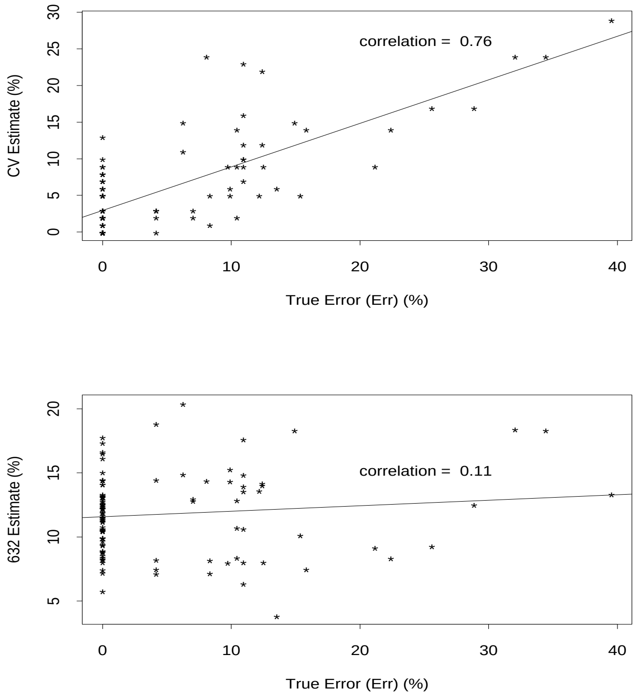

Figure 1: Scatter plot and least-squares line fit for cross-validation and 632 bootstrap estimates versus Err.

was 100 samples. All the experiments gave very similar results so we combined them into two scatter plots. The results are shown in Figure 1. We can see that both cross-validation and 632 bootstrap have high variance as estimators of Err. It is also apparent that Err, the quantity which we are trying to estimate, has high variance as well. FOIL often learns the DNF concepts perfectly from some training sets of a given size, but learns a concept with very high Err on other training sets of the same size. This fact makes the correlation of the estimators with the quantity being estimated of significant interest. The correlation of 632 bootstrap with Err is close to zero, while the correlation of cross-validation with Err is much better, around 0.76. Note that least-squares regression of each estimator on Err gives a line with positive intercept: this indicates that the expected estimate of Err is nonzero even when the target concept is learned perfectly.

The following table shows the mean and (sample) standard deviation of Err and its cross-validation and 632 bootstrap estimates for the 100 runs in the scatter plot experiments.

	Err	*cross-validation*	*632 bootstrap*
dnf1	5.23 ± 8.35	6.04 ± 6.53	11.80 ± 3.24
dnf2	4.57 ± 7.95	4.69 ± 5.42	15.16 ± 3.12
dnf3	0.91 ± 3.08	0.85 ± 1.61	2.89 ± 1.58
dnf4	0.73 ± 2.56	0.94 ± 1.71	3.47 ± 1.41

Clearly 632 bootstrap is biased upward for all four concepts. The bias of cross-validation as an estimator of Err, on the other hand, is very small. The variance of 632 bootstrap is generally much less than that of

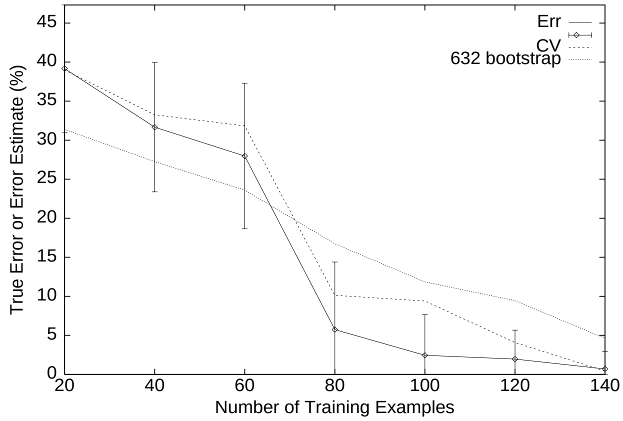

Figure 2: Learning curve for FOIL on dnf1.

cross-validation. This has always been a primary reason for considering bootstrap methods over cross-validation [Efron, 1983].

The scatter plot experiments show the basic pitfalls of cross-validation as a method of estimating Err. The outliers in the scatter plots of cross-validation estimate of Err versus Err itself show that cross-validation estimates *based on a single training set* can greatly over- or under-estimate true error. It is dangerous to conclude that one learned concept definition is better than another solely on the basis of cross-validation. However, the low bias and relatively high correlation of cross-validation with Err indicates that, on average, cross-validation is a good estimator of Err.

Learning curve experiments. To understand better the reasons behind the poor performance of the 632 bootstrap method, and to see how cross-validation performed with different training set sizes, we conducted experiments to construct learning curves for FOIL on the four target concepts.

The learning curve experiments consisted of repeating the basic procedure 10 times for a given training set size and a given distribution to get mean values for Err, cross-validation and 632 bootstrap. This was repeated for various sizes of the training set and plotted. The results are shown in Figure 2 for dnf1. Error bars show plus and minus the sample standard deviations of the measured quantities. For visual clarity, only the error bars for Err are shown.

It can be seen that cross-validation does well over a large range of training set sizes at estimating the mean value of Err for that training set size. Its bias is quite low. On the other hand, the bias of 632 bootstrap is downward for small training sets and upward for large training sets.

The 632 bootstrap estimator has been reported to have lower variance than cross-validation. Figure 3 confirms this: the sample standard deviation (ssd) of 632 bootstrap is lower than that of both cross-validation and Err in almost all cases. For dnf1, the ssd of 632 bootstrap tends to be flat over a wide range of training set sizes. For other target concepts, the ssd of 632 bootstrap

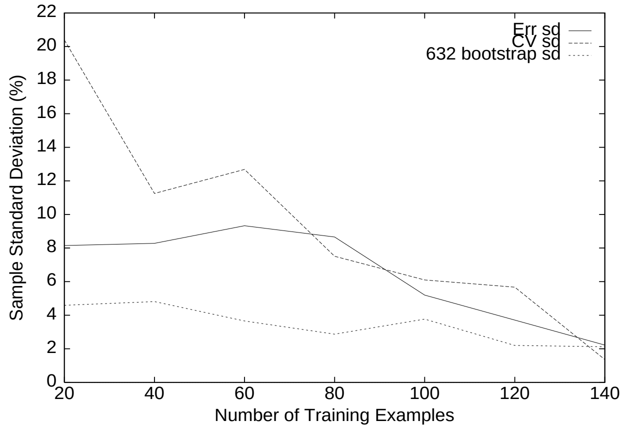

Figure 3: Sample standard deviation of error measures versus training set size for FOIL learning dnf1.

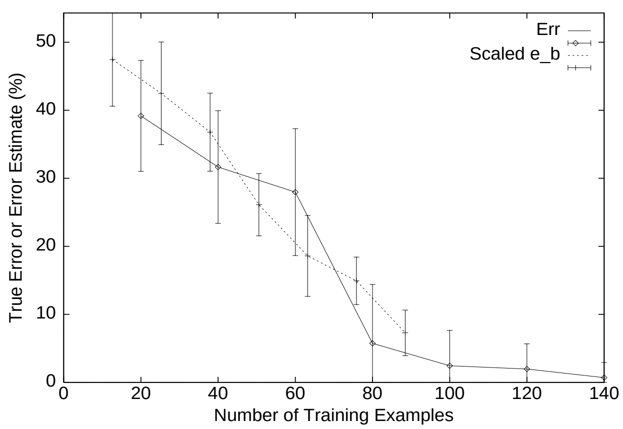

Figure 4: Learning curve for FOIL on dnf1 with bootstrap plotted against 0.632·(number of samples).

becomes larger than that of Err and cross-validation when the size of the training set becomes large and consequently, the value of Err becomes small. The 632 bootstrap method can thus lose its main advantage over cross-validation when the value of Err is small.

Explaining the failure of 632 bootstrap. The poor performance of 632 bootstrap is surprising in view of earlier, positive reports in the statistical literature [Fitzmaurice *et al.*, 1991; Efron, 1983]. However, that work measured the accuracy of 632 bootstrap for training data generated from two multivariate normal classes. The best rule for classifying such data will have nonzero true error since the attributes of an example are not sufficient to determine which class it belongs to. By contrast, our data always came from classes which could be perfectly discriminated.

The 632 bootstrap estimate of error, as mentioned previously, is a weighted average of error e_b on the samples left out during resampling and resubstitution error e_r. The FOIL algorithm tends to learn concepts that "cover" all the training examples. Thus, resubstitution error tends to be close to zero, so the 632 bootstrap estimate of error is essentially equal to $0.632e_b$. It is noticeable that e_b is a good estimate of Err on $0.632n$ samples. This can be seen in Figure 4, which plots the values of e_b measured on n samples at $0.632n$ on the x axis.

Bootstrap resampling results in training sets which are multisets. The expected number of *distinct* points in a resample is about 0.632 times the size of the original dataset.[3] The effect shown in Figure 4 can be explained if FOIL learns essentially the same concept on the bootstrap resampled multiset as it would on the set obtained by removing duplicates from the resample. Figure 5 shows learning curves for FOIL applied to resamples with and without duplicates removed. The results confirm our suspicion. The poor performance of the 632 bootstrap method used with FOIL appears to be due to the fact that FOIL does not benefit from duplicates in a multiset of training examples. The concepts learned by FOIL are, however, different with and without duplicates, as can be seen by the fact that the curves for e_b are different.

It is worth noting that the one-nearest-neighbor classification method, with which the 632 bootstrap method is reported to perform poorly [Weiss, 1991], is a learning algorithm for which by definition duplicates in the training set have no influence. Other learning algorithms with which the 632 bootstrap method has been reported to work well, notably Fisher's linear discriminant method used in [Fitzmaurice *et al.*, 1991] and [Efron, 1983] and the CART decision tree algorithm used in [Crawford, 1989], are strongly influenced by duplicates.

5 Discussion

We studied the performance of cross-validation and 632 bootstrap as estimators of the accuracy of concept definitions learned from synthetic data by FOIL. We observed the following.

(i) The true accuracy Err of learned concepts has high variance. That is, the error of the concept learned by FOIL from independently-drawn sets of examples of the same concept varies widely.

(ii) The 632 bootstrap estimator has lower variance than cross-validation, but it is biased. In our experiments, this bias was upward when the value of Err was below 15%, and downward when Err was above 30%.

(iii) The estimate of Err provided by cross-validation has high variance but is approximately unbiased.

Each of these observations carries implications for future experimental work comparing learning algorithms. The first observation, if it also applies to algorithms other than FOIL, implies that standard statistical tests are invalid for deciding whether one algorithm is significantly better than another, when the available data concerns

[3]The probability of any given example in the original dataset being chosen during resampling is $1-(1-1/n)^n$, which is approximately $1-e^{-1}=0.632$.

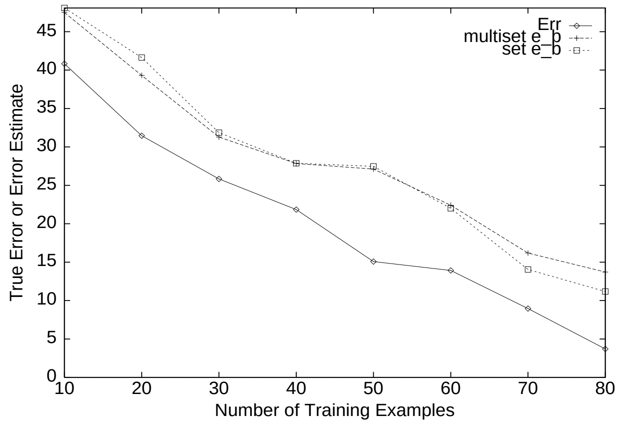

Figure 5: FOIL learns poorly on multisets. Average Err for 10 training sets, multiset resamples and resamples with duplicates removed.

both algorithms learning from single data sets. In [Weinstein *et al.*, 1992], for example, backpropagation in a neural network with 7 hidden units is compared with Fisher's linear discriminant method using a single common data set of 141 examples. The following leave-one-out cross-validation data are given:

	correct	*incorrect*
NN with 7 hidden units	129	12
Linear discriminant	121	20

The authors apply a statistical test similar to the χ^2 test to conclude that backpropagation in a neural network with 7 hidden units performs significantly better than Fisher's linear discriminant method. The null hypothesis in this test is that the concept learned by each algorithm has the same accuracy, and that the observed correct/incorrect numbers are therefore the result of two sets of 141 binomial trials with the same probability of success. However, an extra level of randomness is involved here. The training set used to measure the performance of both algorithms may by chance be one on which backpropagation performs well, whereas the linear discriminant method performs poorly. Despite the significant difference in performance of the two algorithms on *this* data set, the algorithms may well be indistinguishable on independent data sets identically distributed to this data set.

The second observation above is disappointing because previous work has concluded that 632 bootstrap performs better than cross-validation. The poor performance of 632 bootstrap here appears to be caused by the fact that the FOIL algorithm learns almost the same concept on a multiset as on its projection. Future experimental work on learning algorithms should not use the 632 bootstrap method unless all learning algorithms being tested are sensitive to repeated training examples.

Despite the caution required given the first observation listed above, the third observation leads us to recommend continuing to use cross-validation for evaluating the performance of learning algorithms. We recommend cross-validation in particular because of its strong (but not perfect) correlation with Err. However, experimenters must keep in mind that unfortunately, which estimator of learned concept accuracy is best depends on which learning algorithm is used.

References

[Crawford, 1989] Stuart L. Crawford. Extensions to the CART algorithm. *International Journal of Man-Machine Studies*, 31:197–217, 1989.

[Efron, 1979] Bradley Efron. Bootstrap methods: Another look at the jackknife. *Annals of Statistics*, 7:1–26, 1979.

[Efron, 1983] Bradley Efron. Estimating the error rate of a prediction rule: Improvement on cross-validation. *Journal of the American Statistical Association*, 78(382):316–331, June 1983.

[Fitzmaurice *et al.*, 1991] G. M. Fitzmaurice, W. J. Krzanowski, and D. J. Hand. A monte carlo study of the 632 bootstrap estimator of error rate. *Journal of Classification*, 8:239–250, 1991.

[Haussler, 1988] David Haussler. Quantifying inductive bias: AI learning algorithms and Valiant's learning framework. *Artificial Intelligence*, 36:177–221, 1988.

[Kononenko and Bratko, 1991] Igor Kononenko and Ivan Bratko. Information-based evaluation criterion for classifiers' performance. *Machine Learning*, 6(1):67–80, January 1991.

[Quinlan, 1990] John R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

[Sanchez and Cepeda, 1989] J. M. Prada Sanchez and X. L. Otero Cepeda. The use of smooth bootstrap techniques for estimating the error rate of a prediction rule. *Communications in Statistical Simulation*, 18(3), 1989.

[Towell *et al.*, 1990] G. G. Towell, J. W. Shavlik, and Michiel O. Noordewier. Refinement of approximate domain theories by knowledge-based artificial neural networks. In *Proceedings of the National Conference on Artificial Intelligence*, pages 861–866, 1990.

[Weinstein *et al.*, 1992] John N. Weinstein *et al.* Neural computing in cancer drug development: Predicting mechanism of action. *Science*, 258:447–451, October 1992.

[Weiss and Kulikowski, 1991] Sholom M. Weiss and Casimir A. Kulikowski. *Computer systems that learn: Classification and prediction methods from statistics, neural nets, machine learning, and expert systems.* Morgan Kaufmann Publishers, Inc., 1991.

[Weiss, 1991] Sholom M. Weiss. Small sample error rate estimation for k-NN classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):285–289, March 1991.