Machine Learning, 24, 1–3 (1996)

Editorial

THOMAS G. DIETTERICH tgd@cs.orst.edu
Executive Editor

This issue contains a paper by Leo Breiman ("Bagging Predictors") that has important implications for many aspects of machine learning. Breiman observes that the decision-tree classifier produced by his well-known CART algorithm (Breiman, Friedman, Olshen, & Stone, 1984) can vary substantially when a small number of training examples are added or deleted from the training set. He terms this "instability," and he points out that many other learning algorithms, including Quinlan's C4.5 algorithm (Quinlan, 1993), variable subset selection methods, and the error backpropagation algorithm are all unstable in this sense. This instability affects not only the structure of the decision trees but also the classification decisions made by the trees. In short, two runs of CART on slightly different data sets will typically disagree on the classifications of many test-set cases.

In this editorial, I want to briefly point out six important implications of this observation.

The first implication concerns the notion of "bias" as it has been defined and explored in our field. Many researchers have set as an important goal for the field to find "biases" that work well for a broad range of learning tasks (preferrably learning tasks encountered in the "real world"). This has usually been conceived of as identifying a space of hypotheses that contains hypotheses that closely approximate target functions (or target concepts) in the "real world." However, Breiman's paper demonstrates that a significant portion of the classification errors of our learning algorithms are not the result of searching in the wrong hypothesis space—at least in the sense that the space does not contain a good approximation to the target function. Instead, the errors are the result of instability in the search process. This suggests that our worries about identifying the "right" hypothesis space are somewhat misplaced. We may find the right hypothesis space and yet still obtain high error rates.

The second implication of Breiman's work is that it suggests a research path that may lead to improved general-purpose learning algorithms. Specifically, it suggests that we should identify sources of instability and eliminate them.

What are the sources of instability? The most fundamental source of instability is that the hypothesis space is too large. If an algorithm searches a very large hypothesis space and outputs a single hypothesis, then in the absence of huge amounts of training data, the algorithm will need to make many arbitrary decisions—decisions whose outcome might be different if the training set were slightly different. This is "informational" instability—instability caused by the lack of information.

However, some of our learning algorithms also suffer from "computational instability". For example, when backpropagation is initialized with small random weights, these random weights introduce huge initial instability in the search (Kolen & Pollack, 1991). If this instability could be removed without changing the space of learnable functions, the performance of backprop should improve.

Sometimes it is difficult to separate informational and computational instability. It appears that an important source of instability in the top-down decision tree algorithms is the fact that the algorithms must choose a single split at each node. That split decision is often made based on nearly-equal values of the splitting heuristic (i.e., nearly equal GINI scores or gain ratios). The addition or deletion of a single training example could alter the decision. Perhaps there are ways of stabilizing this decision that could improve the algorithms?

Obviously another way of reducing instability would be to shrink the hypothesis space. However, this would also reduce the range of problems that could be solved using our learning algorithms. If intelligent agents need broadly-applicable learning algorithms, then they will need some good ways of controlling instability without over-constraining the space of hypotheses. In short, our representations should be general, but our algorithms should still be stable.

The third important implication of Breiman's paper is really the converse of the previous point. Sometimes, researchers have shown that a change in an algorithm produces improved performance, as measured experimentally on many data sets. Sometimes the researchers have concluded that this improvement is the result of such things as improved splitting criteria, improved pruning procedures, improved feature selection, improved constructive induction, and so on. However, Breiman's paper suggests that some of these improvements may have resulted merely from an arbitrary stabilization of the algorithm. For example, the decision-tree algorithms could be stabilized simply by always preferring the feature having the lowest index when there is a near-tie in the splitting criterion. Authors need to consider these kinds of stabilizers as null hypotheses and show through careful experimentation that performance has improved for the claimed reasons.

The fourth important implication of Breiman's paper concerns the statistical comparison of learning algorithms. Because of the instability of learning algorithms, any comparison of two different algorithms must involve making multiple runs of the algorithms in order to remove this source of variation from the comparison. In particular, note that even if the test set is very large, performance of a single run of an algorithm measured on that test set only tells us (with high precision) the performance of that algorithm on that single run. In some of my own papers (e.g., Dietterich, Hild, & Bakiri, 1995), we conducted only one run of each algorithm and then applied a test for the difference of two proportions to draw statistical conclusions. We did not consider the possibility that if the algorithms were run again on a second training set, the results could have been very different.

A fifth consequence of Breiman's paper is that it raises a challenge for the theoretical study of machine learning. Is it possible to predict the degree of instability of a learning algorithm by analyzing the algorithm itself? Can we characterize in a formal way the sources of computational instability and thereby provide ways of removing them? Is it possible to forge a link between the average-case bias/variance framework of statistics and the various formal models developed in machine learning?

The final implication of Breiman's paper is particularly relevant to applications of machine learning for knowledge discovery. An important goal of machine learning research is to develop algorithms that can produce understandable hypotheses. Our users want to read the decision trees or rules produced by our algorithms and "believe" them. Because our algorithms are unstable, it is critical to remind ourselves and our users that the rules or trees produced by our algorithms could change very substantially if the training set changed slightly. (Indeed, such experiments should be conducted so that we can present our users with a range of possible hypotheses rather than a single hypothesis.) In short, extreme caution must be exercised when interpreting the outputs of learning algorithms.

References

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and Regression Trees.* Wadsworth International Group.

Dietterich, T. G., Hild, H., & Bakiri, G. (1995). A comparison of ID3 and backpropagation for English text-to-speech mapping. *Machine Learning, 18*, 51–80.

Kolen, J. F., & Pollack, J. B. (1991). Back propagation is sensitive to initial conditions. In *Advances in Neural Information Processing Systems*, vol 3, pp. 860–867 San Francisco, CA. Morgan Kaufmann.

Quinlan, J. R. (1993). *C4.5: Programs for Empirical Learning.* Morgan Kaufmann, San Francisco, CA.