Machine Learning, 6, 205–206 (1991)

A Reply to Zito-Wolf's Book Review of *Learning Search Control Knowledge: An Explanation-Based Approach*

STEVEN MINTON
NASA Ames Research Center, Mail Stop 244-17, Moffett Field, Mountain View, CA 94035

I would like to thank Roland J. Zito-Wolf, not only for his careful, thorough review of my book, but also for pointing out several confusing issues that I can now take the opportunity to clarify.

According to Zito-Wolf, my book suggests that learning from failure is inherently more effective than learning from success. I would say that the book's claims are actually a bit weaker. It is true that in the experiments I conducted, learning from failure was generally more useful than learning from success (regardless of whether the knowledge learned from success was encoded as control rules or traditional macro-operators.) However, these results do not warrant the conclusion that learning from failure is *inherently* better. Indeed, as the review mentions, we know that the utility of a learning method depends on a variety of factors, including the domain and how the learned information is used. The main conclusion drawn from the PRODIGY/EBL work is that the learning method should be sensitive to its effect on the problem-solving architecture. So, while it is true that learning from failure can produce useful rules such as "Don't try to stack a block on itself," there are also times when learning from success can produce macros that, like scripts, can be frequently reused, and are correspondingly useful. A learning method should have the flexibility to learn from different types of experiences, and to make use of that knowledge effectively.

Zito-Wolf also points out an area of potential confusion regarding the speedups produced by PRODIGY's EBL component. As he states in his review, the rationale for learning is the exponential worst-case cost of uninformed search. Zito-Wolf questions why the exponential cost does not show up in the experimental results. Specifically, the time taken by PRODIGY without learning does not appear to rise exponentially with increasing problem size, and thus, the overall speedups described in the "Performance Results" section are not as impressive as one might expect. This is especially puzzling, Zito-Wolf points out, because on some of the example problems discussed earlier in the text, uninformed problem solving does take exponential time and consequently learning can achieve dramatic speedups. The explanation for this apparent discrepancy comes from the fact that measuring the overall performance on *large collections of problems* is very different than characterizing the system's asymptotic performance on a particular type of example. Most notably, when I was testing large collections of problems, practical considerations required that the system be given a constant time bound after which problem-solving was terminated. Under these experimental conditions, the worst-case performance of the system is actually constant! This explains why the graphs accompanying the experimental results do not show an exponential rise

in problem-solving time for uninformed search.[1] On the larger problems, the uninformed problem solver would frequently hit this time bound. Unfortunately, the book does not contain the tables from my thesis which illustrate this point, a shortcoming which I regret.

The review also states that PRODIGY's reliance on domain-dependent axioms in its COMPRESSION module is not quantified. Actually, my book does address this issue, albeit briefly, and only for the blocksworld domain. Experimental results show that if the domain-dependent axioms are left out, PRODIGY's overall performance after learning does suffer by about 30 percent, but performance is still significantly better than without learning. (Just one-third of the learned rules benefited from the domain-specific axioms; these rules were reduced in size by an average of 19 percent). The chapter on "Compression" discusses why this extra domain-specific information is necessary.

Finally, I am grateful for Zito-Wolf for his supportive comments regarding my effort to precisely characterize EBL. Given the close relationship between EBL and logic program optimization, he suggests that in the future we should take advantage of the work in logic programming, and I am in complete agreement. The PRODIGY/EBL work took place between 1984 and 1988 (when the book was published), and at that time, the close relationship between these two approaches was not fully appreciated.

The purpose of my work with the PRODIGY/EBL system was to study the strengths and limitations of EBL. In my opinion, the primary contribution of the book is the description of the utility problem and techniques for dealing with the utility problem. In order to facilitate further research in this area, I would like to mention that the PRODIGY system (with a manual) is available by contacting the author, or sending electronic mail to PRODIGY@cs.cmu.edu. The experiments described in the book can all be easily replicated using the software provided.

Note

1. There is a secondary factor which is also relevant: the problems are not *necessarily* increasing in difficulty. As discussed in the book, test problems were randomly generated with increasing *maximum* problem size. Thus some of the problems generated later were actually smaller in size. (In retrospect, this was probably a bad decision on my part.) Secondly, large problems are not necessarily hard, although the worst-case performance is exponential with increasing size.