Learning to Teach with a Reinforcement Learning Agent

Joseph E. Beck
Department of Computer Science
University of Massachusetts, Amherst
Amherst, MA 01003. U.S.A.
beck@cs.umass.edu

Intelligent tutoring systems (ITS) use artificial intelligence techniques to customize their instruction to fit the needs of each student. To do this, the system must have knowledge of the student being taught (commonly called a *student model*) and a set of pedagogical rules that enable the system to follow good teaching principles. Teaching rules are commonly represented as a set of "if-then" production rules, where the "if" side is dependent on the student model, and the "then" side is a teaching action. For example, a rule may be of the form "IF (the student has never been introduced to the current topic) THEN (teach the topic to the student)."

This approach is fairly straightforward from a knowledge engineering perspective, but has many drawbacks. First, there are many such rules, and it is very expensive to encode all of the rules the system will require to teach effectively. Second, it is difficult to incorporate knowledge that human tutors do not use themselves, since expert teachers cannot describe how the system should reason with such knowledge. Since machine tutors have a very different set of data available than human tutors (keystroke latencies, performance on a similar problem 3 weeks ago, etc.), knowledge that could dramatically improve the tutor's performance must be ignored.

We will use Reinforcement learning (RL) to train ITS to teach. RL is a mechanism that allows a machine learning (ML) agent to learn given only a description of the state and a reward for its performance. Thus RL supports unsupervised learning as it does not need to be told what the correct action is, but simply learns how to maximize its long term reward.

We will use the student model as a state description for the RL-agent. This saves considerable knowledge engineering work. Also, since the student model already accounts for student learning, the learning task becomes *stationary*. That is, for a given set of inputs the expected value will remain constant. The agent uses this state description as input to a function approximator, and uses this learned function to select a teaching action. After the selected action is performed, the system examines how the student performs on future problems to compute the effectiveness of the teaching action. If the teaching action was effective, student performance should improve, and the system will receive a positive reward. Less appropriate teaching actions will not be as effective at improving a student's performance, and will receive smaller rewards. We propose that a state-action learning algorithm such as SARSA(Sutton & Barto 1997), a on-line version of Q-learning, would be most applicable for this task. The system learns to map each student and his current knowledge to an optimal teaching action. For example, the system could learn when is the best time to present an example to the student, or when a topic should be taught, or when the student should be given a problem to solve.

It is unlikely the system will be able to learn this policy quickly enough from a single student. Therefore, the RL-agent will first be trained off-line using either empirical data, or simulations of students(VanLehn, Ohlsson, & Nason 1996). Once the system attains a reasonable level of teaching ability, it can be used by actual students. The learning mechanism will be left in place, so the system can fine-tune its performance to better fit with each student. This permits a level of customization that is impossible with precanned teaching rules.

Such a system would advance research in both ITS and ML. ITS would benefit from having more flexible teaching rules. These rules would be customized to each individual learner, and would be able to consider knowledge that is (at best) problematic to add when working with human experts. This represents a fundamental shift in system construction, and would potentially both reduce costs and provide great improvements in learning gains exhibited by the tutor's users.

This methodology explores how to train an ML system using a large quantity of poorer quality data (the empirical studies and simulations), and then fine-tune it with higher quality, but more scarce data (the interactions with the student using the system). This permits ML systems to be used in domains where they would normally be undeployable due to a scarcity of data.

References

Sutton, R., and Barto, A. 1997. *An Introduction to Reinforcement Learning.* MIT Press.

VanLehn, K.; Ohlsson, S.; and Nason, R. 1996. Applications of simulated students: An exploration. *Journal of Artificial Intelligence in Education* 5(2):135–175.