Improving Reinforcement Learning Function Approximators via Neuroevolution

Shimon Whiteson
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712 USA
shimon@cs.utexas.edu
http://www.cs.utexas.edu/~shimon

Abstract

Reinforcement learning problems are commonly tackled with *temporal difference methods*, which use dynamic programming and statistical sampling to estimate the long-term value of taking each action in each state. In most problems of real-world interest, learning this value function requires a *function approximator*, which represents the mapping from state-action pairs to values via a concise, parameterized function and uses supervised learning methods to set its parameters. Function approximators make it possible to use temporal difference methods on large problems but, in practice, the feasibility of doing so depends on the ability of the human designer to select an appropriate representation for the value function. My thesis presents a new approach to function approximation that automates some of these difficult design choices by coupling temporal difference methods with policy search methods such as evolutionary computation. It also presents a particular implementation which combines NEAT, a neuroevolutionary policy search method, and Q-learning, a popular temporal difference method, to yield a new method called NEAT+Q that automatically learns effective representations for neural network function approximators. Empirical results in a server job scheduling task demonstrate that NEAT+Q can outperform both NEAT and Q-learning with manually designed neural networks.

Thesis Overview

In many machine learning problems, an agent must learn a *policy* for selecting actions based on its *state*, which consists of its current knowledge about the world. *Reinforcement learning* problems are the subset of these tasks for which the agent must learn a policy without ever seeing examples of correct behavior. Instead, it receives only positive and negative feedback for the actions it tries. Since many practical, real world problems (such as robot control, game playing, and system optimization) fall in this category, developing effective reinforcement learning algorithms is critical to the progress of artificial intelligence.

The most common approach to reinforcement learning relies on *temporal difference methods* (Sutton & Barto 1998), which use dynamic programming and statistical sampling to estimate the long-term value of taking each possible action in each possible state. Once this value function has been learned, an effective policy can be trivially derived. For small problems, the value function can be represented in a table, with one entry for each possible state-action pair. However, for most problems of real-world interest, the tabular approach is infeasible because the agent would never have a chance to visit every state much less learn the correct value for each state-action pair. In such cases, temporal difference methods are coupled with a *function approximator* which represents the mapping from state-action pairs to values via a more concise, parameterized function and uses supervised learning methods to set its parameters. Many different methods of function approximation have been used successfully, including CMACs, radial basis functions, and neural networks (Sutton & Barto 1998).

Function approximators make it possible to use temporal difference methods on problems with large state and action spaces. However, in practice, the feasibility of doing so depends largely on the ability of the human designer to select an appropriate representation for the function approximator (e.g. the topology and initial weights of the neural network). Unfortunate design choices can result in estimates that diverge wildly from the optimal value function (Baird 1995) and agents that perform extremely poorly.

The primary contribution of this thesis is a new approach to doing function approximation that automates some of these difficult design choices. It does so by coupling temporal difference methods with policy search methods such as evolutionary computation. In particular, I use NeuroEvolution of Augmenting Topologies (NEAT) (Stanley & Miikkulainen 2002), a method that uses evolutionary computation to learn both the topology and weights of neural networks, in conjunction with Q-learning (Watkins 1989), a popular temporal difference method. The resulting method, called NEAT+Q, uses NEAT to learn the topology and initial weights of networks which are then updated, via backpropagation (Bishop 1995), towards the value estimates provided by Q-learning.

In addition to automatically learning appropriate network topologies and initial weights for Q-learning, NEAT+Q also makes it possible to take advantage of the Baldwin Effect, a phenomenon whereby populations whose individuals learn during their lifetime adapt more quickly than populations whose individuals remain static (Baldwin 1896). In the Baldwin Effect, which has been demonstrated in evolutionary computation (Ackley & Littman 1991; Boers, Borst, & Sprinkhuizen-Kuyper 1995), evolution proceeds more quickly because an individual does not have to be exactly

right at birth; it need only be in the right neighborhood and learning will adjust it accordingly. By combining learning *across* fitness evaluations with learning *within* fitness evaluations, NEAT+Q has the potential to reap the Baldwin Effect.

This thesis presents empirical results from the domain of server job scheduling, a challenging reinforcement learning task from the burgeoning field of *autonomic computing* (Kephart & Chess 2003). My experiments demonstrate that NEAT+Q, by automatically discovering appropriate topologies and initial weights, can dramatically outperform a Q-learning approach that uses manually designed neural networks. These experiments also demonstrate that when NEAT is used by itself (i.e. to learn policies directly without estimating value functions), it does not perform as well as NEAT+Q, which harnesses the power of temporal difference methods.

Additional Contributions

In addition to the novel method described above for synthesizing temporal difference and policy search methods, this thesis presents the following additional contributions: 1) two novel enhancements to policy search methods that improve their on-line performance by borrowing exploratory mechanisms from temporal difference methods, and 2) a new method called FS-NEAT that uses neuroevolution to automate the task of feature selection in reinforcement learning. The following subsections provide a brief overview of these contributions.

On-Line Policy Search

Reinforcement learning agents are typically trained using policy search methods or temporal difference methods. While there is much debate about the efficacy of policy search methods, there are some problems on which they have achieved the best results to date (Bagnell & Schneider 2001; Kohl & Stone 2004). However, policy search methods do not fare well in *on-line* scenarios, in which there are real-world consequences for the agent's behavior during learning. They do not excel in such scenarios because they lack mechanisms for balancing the need to search for better policies (exploration) with the need to accrue maximal reward (exploitation). This thesis presents two novel enhancements to policy search methods that improve their on-line performance by borrowing exploratory mechanisms from temporal difference methods. The first modification, which can be applied to any policy search method, borrows the notion of ϵ-greedy exploration, resulting in *ϵ-greedy policy search*. This algorithm switches probabilistically between searching for better policies and re-evaluating the best known policy to garner maximal reward. The second modification, which requires a population-based policy search method such as a genetic algorithm (Goldberg 1989), borrows the notion of softmax selection, resulting in *softmax policy search*. It distributes evaluations in proportion to each individual's estimated fitness, thereby focusing on the most promising individuals and increasing the average reward accrued. I compare the resulting methods in two domains: elevator control and auto racing. The results demonstrate that these new techniques significantly improve the on-line performance of NEAT, a neuroevolutionary policy search method.

Automatic Feature Selection via Neuroevolution

Feature selection is the process of finding the set of inputs to a machine learning algorithm that will yield the best performance. Developing a way to solve this problem automatically would make current machine learning methods much more useful. Previous efforts to automate feature selection rely on expensive meta-learning or are applicable only when labeled training data is available. This thesis presents a novel method called FS-NEAT which extends the NEAT (Stanley & Miikkulainen 2002) neuroevolution method to automatically determine an appropriate set of inputs for the networks it evolves. By learning the network's inputs, topology, and weights simultaneously, FS-NEAT addresses the feature selection problem without relying on meta-learning or labeled data. Initial experiments in an autonomous car racing simulation demonstrate that FS-NEAT can learn better and faster than regular NEAT. In addition, the networks it evolves are smaller and require fewer inputs. Furthermore, FS-NEAT's performance remains robust even as the feature selection task it faces is made increasingly difficult.

References

Ackley, D., and Littman, M. 1991. Interactions between learning and evolution. *Artificial Life II, SFI Studies in the Sciences of Complexity* 10:487–509.

Bagnell, J., and Schneider, J. 2001. Autonomous helicopter control using reinforcement learning policy search methods. In *Proceedings of the International Conference on Robotics and Automation 2001*. IEEE.

Baird, L. C. 1995. Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the Twelfth International Conference on Machine Learning*, 30–37. Morgan Kaufmann.

Baldwin, J. M. 1896. A new factor in evolution. *The American Naturalist* 30:441–451.

Bishop, C. M. 1995. *Neural Networks for Pattern Recognition*.

Boers, E.; Borst, M.; and Sprinkhuizen-Kuyper, I. 1995. Evolving Artificial Neural Networks using the "Baldwin Effect". Technical Report TR 95-14.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*.

Kephart, J. O., and Chess, D. M. 2003. The vision of autonomic computing. *Computer* 36(1):41–50.

Kohl, N., and Stone, P. 2004. Machine learning for fast quadrupedal locomotion. In *The Nineteenth National Conference on Artificial Intelligence*, 611–616.

Stanley, K. O., and Miikkulainen, R. 2002. Evolving neural networks through augmenting topologies. *Evolutionary Computation* 10(2):99–127.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Watkins, C. 1989. *Learning from Delayed Rewards*. Ph.D. Dissertation, King's College, Cambridge.