Mitigating Catastrophic Forgetting in Temporal Difference Learning with Function Approximation

Benjamin Goodrich
Department of Electrical Engineering and Computer Science
University of Tennessee
Knoxville, TN 37996
bgoodric@utk.edu

Itamar Arel
Department of Electrical Engineering and Computer Science
University of Tennessee
Knoxville, TN 37996
itamar@ieee.org

Abstract

Neural networks have had many great successes in recent years, particularly with the advent of deep learning and many novel training techniques. One issue that has prevented reinforcement learning from taking full advantage of scalable neural networks is that of catastrophic forgetting. The latter affects supervised learning systems when highly correlated input samples are presented, as well as when input patterns are non-stationary. However, most real-world problems are non-stationary in nature, resulting in prolonged periods of time separating inputs drawn from different regions of the input space.

Unfortunately, reinforcement learning presents a worst-case scenario when it comes to precipitating catastrophic forgetting in neural networks. Meaningful training examples are acquired as the agent explores different regions of its state/action space. When the agent is in one such region, only highly correlated samples from that region are typically acquired. Moreover, the regions that the agent is likely to visit will depend on its current policy, suggesting that an agent that has a good policy may avoid exploring particular regions. The confluence of these factors means that without some mitigation techniques, supervised neural networks as function approximation in temporal-difference learning will only be applicable to the simplest test cases.

In this work, we develop a feed forward neural network architecture that mitigates catastrophic forgetting by partitioning the input space in a manner that selectively activates a different subset of hidden neurons for each region of the input space. We demonstrate the effectiveness of the proposed framework on a cart-pole balancing problem for which other neural network architectures exhibit training instability likely due to catastrophic forgetting. We demonstrate that our technique produces better results, particularly with respect to a performance-stability measure.

Keywords: Catastrophic Forgetting, Neural Networks, Reinforcement Learning

1 Introduction

Catastrophic forgetting is a known phenomenon in supervised neural network settings as well as other parameterized learning systems. One issue that prompts the problem of forgetting is correlated samples, which tend to produce gradients of persistent direction. This distorts the parameters of the network, thereby discarding previously learned representations.

In the case of offline training, where one has all of the data available in advance, input examples are usually shuffled. This effectively results in a stationary distribution for the input samples. That is, each example is viewed as drawn independently from a fixed distribution. In online learning, however, one does not have all of the data in advance. The data can only be drawn as it is presented by the environment. If the data is being drawn in a correlated, or non-stationary manner, then this presents major training difficulties for neural networks

With temporal-difference learning in particular, the input distribution is non-stationary for a variety of reasons. Short term correlations occur since each successive state transition will produce a state that is connected to the previous sampled state. This alone can cause problems when training with a neural network approximating the value function, since the network will receive value updates for a sequence of closely related inputs. The input distribution can change over longer time-scales as well since an agent can only be present in one region of the environment at any given time interval, and may spend substantial time in a relatively small number of regions.

This problem is not uncommon since a good policy usually will preclude the agent from visiting certain regions of the state/action space. Consider a near optimal policy for a given MDP. Under many environments, a near optimal policy will dictate that the agent avoid regions that lead to failure. Should a neural network learn an optimal policy, it may quickly forget how to maintain such policy due to lack of visitations at failure regions.

2 Background

Catastrophic forgetting is a well studied phenomenon. In the context of reinforcement learning it has been recently recognized by various researchers, and is at times referred to as the unlearning problem [1] [2] [3].

In a more recent success of applying reinforcement learning combined with deep neural networks, an agent was trained to play Atari video games [4]. One major contribution to the success of this work, was the utilization of a "replay buffer." This replay buffer helps mitigate catastrophic forgetting by storing off-policy experience of the agent and sampling from it randomly when performing updates to the network.

While a replay buffer can somewhat resolve the problem of correlations in training examples, there are issues that it may not be able to overcome. As mentioned in the previous section, an agent that is learning a near-optimal policy may stop visiting certain regions of the state space altogether. If this occurs, then the replay buffer of a fixed size will eventually discard any experience gained from visiting those regions. One could scale a replay buffer by increasing its size at the expense of memory, however scaling in this manner entirely contradicts the very reason neural networks are used to learn in the first place. Neural networks are meant to approximate a mapping without requiring the storing of all the data in the first place. If one has enough memory to store all of the data, then a neural network may not be the best tool for learning. Instead, it may be more appropriate to use other memory-based learning techniques such as localized linear regression [5].

Clearly, catastrophic forgetting mitigation strategies are necessary for facilitating scalability in a more data-driven manner than a replay buffer. Ideally, new neural network architectures should be explored that are inherently designed to address non-stationary input streams, as well as retain representations from prior training trajectories that have not been visited over long periods of time.

3 Neuron Selection Technique

The technique proposed here, dubbed "cluster-select", has recently been introduced as a powerful technique applied to non-stationary classification tasks [6] [7]. In this work, we extend this framework and apply it to online reinforcement learning.

The primary motivation for our work is in the trade-off between techniques that use fully global representations and those that employ strictly local representations. Most neural network activation functions are nonzero for most of the input space, which yield global representations. This means that an update to minimize the error in one region of the input space will affect distant, unrelated regions, likely increasing any error in those regions. Machine learning techniques that employ fully global representations tend to generalize well yet suffer from catastrophic forgetting.

On the other hand, some machine learning techniques employ fully local representations. This would include techniques such as tabular methods, memory-based learning techniques, and radial basis functions. These techniques do not suffer from catastrophic forgetting, however they all have trouble generalizing, because learning something at one region can not be applied to other regions of the input space. Additionally, they all suffer from the curse of dimensionality since for every added dimension, more storage is required. For tabular methods, this manifests as exponentially more memory required with each added dimension. Likewise, radial basis representations will need exponentially more centroids for a representation to effectively cover a high-dimensional input space.

For cluster-select we seek to find a balance between fully global and fully local techniques. There should exist techniques that have some generalization power of global techniques, but also have some properties of local techniques in that they do not suffer from catastrophic forgetting.

With regular feed-forward networks, neuron j can be viewed as having an input weight vector $\vec{w_j}$ associated with it. Cluster-select adds a centroid vector $\vec{c_j}$ in addition to this weight vector. During the feed forward pass, the distance between the input vector $\vec{x}$ and the centroid j is computed using Euclidean norm, such that

$$d_j = \|\vec{x} - c_j\|_2^2 \tag{1}$$

This vector of distances is used to select the k neurons that are nearest to the centroid. Only the k nearest neurons are allowed to have a nonzero output, and all others have their value forced to 0. If an output y_j for a neuron j is allowed to be nonzero, it is computed in the standard way that is done for feed forward neural networks: $y_j = f(\vec{w_j} \cdot \vec{x})$ or simply the dot product of the weights and input (with a bias term assumed to be included as an input), and some nonlinear activation function $f(\cdot)$.

Back-propagation is thus only allowed to occur along the path of the k nearest neurons, with the error derivative forced to 0 for all other neurons. This effectively builds a localized sub-network for each region of the input space, which shares neurons with nearby regions. Effectively, this adds a localized representation to a neural network.

4 Experimental Results

For our test, we considered the the classic cart-pole reinforcement learning problem with no friction. The equations governing the dynamics of this problem can be found in [8]. The problem involves a simulated cart on a horizontal track, with a pole attached to it. The action space has been discretized such that a total of 3 actions involve applying a left force, right force, or no force. This essentially produces bang-bang controls. An episode consists of the cart with the pole started from a random angle, and a small random velocity. An episode proceeds until one of the state variables either grows too large (within reasonable bounds), or 1000 steps elapsed.

Keeping the state variable within reasonable bounds meant that the cart horizontal position and velocity, as well as the pole angular position and velocity were all limited. This was necessary, since a system having no friction would mean that these state variables could be unbounded, potentially producing strange behavior if they grew too large. A negative reward was assigned if the episode ended prematurely due to one of the state variables becoming out of bounds. The goal of the task is to balance the pole upright by applying the horizontal forces to the cart, hence a small positive reward was applied for every frame that the pole was in an upright position.

Each step of the system was simulated using the Runge-Kutta method of numerically solving the differential equations that describe the system. Each step in the simulation consisted of roughly 20.0 milliseconds of simulated time, such that 50 steps equals one second. These tests all used the SARSA learning algorithm with one temporal difference update performed on the network for every step (no batch updates, or replay buffers were used).

To test each method, a random search was performed over hyper-parameters. Hyper-parameters generally included: the learning rate, a small decay constant for the learning rate to decay, the number of hidden neurons, the gamma constant for temporal difference learning, the amount of reward to provide the agent for balancing the pole relative to the amount of negative reward for going out of bounds, the initial ϵ to use for ϵ-greedy exploration, the amount to decay ϵ. For cluster-select, there was an additional hyper-parameter for the number of neurons to select for a feed forward pass.

Each activation function was examined separately, where reasonable selections for the hyper-parameters were provided for the random search. Upon performing approximately 200 runs for each activation function with a given set of hyper-parameters, those that produced the best results were selected. Note that in the plots, performance was measured as a function of the total number of steps that the agent was able to balance the pole and collect reward for an episode. The total reward was not plotted, because the amount of reward to assign was a hyper-parameter.

Figure 1 provides results for a simple case with tabular value function. This particular result had its hyper-parameters hand-tuned (i.e. no random searches of hyper-parameters were performed), and it is provided as a simple baseline level of performance for the tabular case. The value function was maintained in a table of 80,000 states where the entry in the

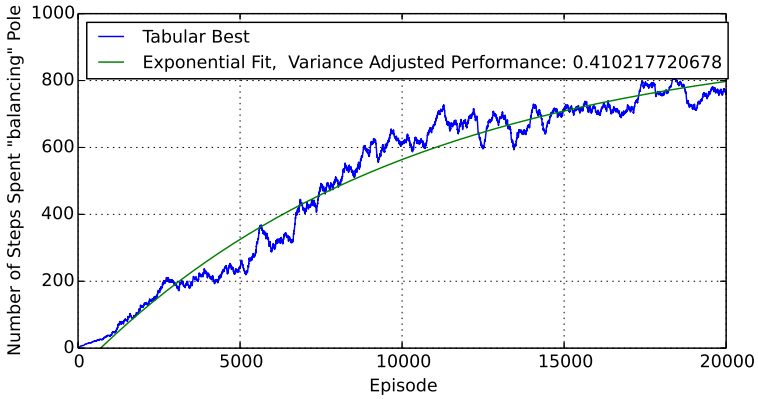

Figure 1: Result for a Tabular $Q_{s,a}$ Estimator

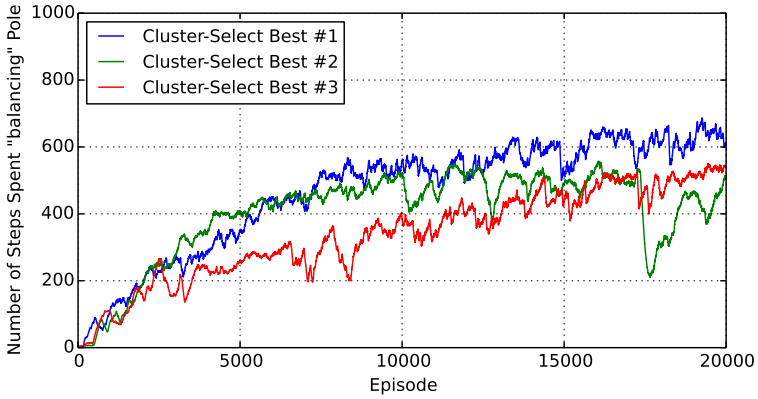

Figure 2: Result for Cluster-Select Neural Net $Q_{s,a}$ Estimator

table was obtained as a function of the 4 state variables. This was done by binning the state variables such that the cart position had 10 bins; the cart velocity, pole angle, and pole angular velocity all had 20 bins. The binning was performed over the valid ranges of these state variables. The number of bins for each state variable was a hyper-parameter which we hand-tuned. Figure 1 also shows a fit to an exponential curve of the form $f(x) = a - b\exp(-cx)$ where a, b, c are constants pertaining to the fitted curve.

Figure 3 illustrates the results for a neural network with linear rectified activation functions. This activation function produced some agents with the best performance. Unfortunately the good performance was unstable, and would often regress as shown in this figure. These agents would learn to balance the pole well, then suddenly regress to terrible performance. We hypothesize that this sudden regression is caused by catastrophic forgetting in the hidden layers. Essentially, after the agent begins to learn to balance the pole well, it is unable to maintain this policy since the network is no longer being trained on the failure states. Eventually it drops the pole, and 'unlearns' the previous captured representation. A plot of performance for networks with sigmoid and hyperbolic tangent activations is not provided, since these networks did not reach adequate levels of performance. It is unclear why these particular activation functions failed to deliver a proficient policy. It is possible that they simply required more training time, or that a good set of hyper-parameters was never found. It is also possible that these activation functions are a poor match for this particular problem.

On the other hand, the cluster-select technique generally had a much smoother learning curve. In particular, the learning profile does not exhibit sudden dips (regressions) in performance, as Figure 2 clearly illustrates. In addition, Table 1 provides an objective measure of performance expressed as the log of the variance-adjusted performance. To compute the latter we first fit the performance curve to an exponential function, as depicted in Figure 1. Next, we measure the mean squared deviation of the original learning curve from the fitted function. Finally, we define the variance-adjusted performance as the mean of the squared values of the original learning curve relative to the mean squared deviation from the fitted function. This metric favors a learner that is both stable in its learning profile as well as reaches a high performance level.

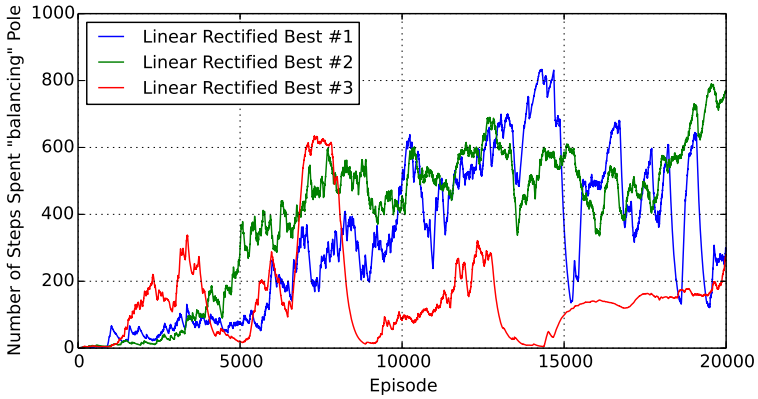

Figure 3: Result for a Linear Rectified Neural Net $Q_{s,a}$ Estimator

	Tabular	Hyperbolic Tangent	Sigmoid	Linear Rectified	Cluster-Select
Performance (Mean Squared Sum)	310311	3674	17682	202616	243417
Deviation from Exponential Fit	1944	225.4	1083	7618	801.6
Log Variance Adjusted Performance	5.073	2.791	2.793	3.281	**5.716**

Table 1: Summary of Results

5 Conclusion

This work has demonstrated an unlearning effect that neural networks exhibit when combined with reinforcement learning. This was illustrated by establishing a test case that exhibits this phenomenon. A neural network architecture was proposed that aims to reduce catastrophic forgetting by localizing activations of the network to regions of the input space. This proposed architecture was shown to substantially reduce this unlearning effect.

We hope that the broader impact of this work will bring attention to this problem of catastrophic forgetting, as more work needs to be put into developing solutions in order for the reinforcement learning community to benefit from modern advances in neural networks and deep learning.

References

[1] V. U. Cetina, "Multilayer perceptrons with radial basis functions as value functions in reinforcement learning." in *ESANN*, 2008, pp. 161–166.

[2] S. Weaver, L. Baird, and M. Polycarpou, "Preventing unlearning during online training of feedforward networks," in *Intelligent Control (ISIC), 1998. Held jointly with IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA), Intelligent Systems and Semiotics (ISAS), Proceedings*. IEEE, 1998, pp. 359–364.

[3] J. R. N. Forbes, "Reinforcement learning for autonomous vehicles," Ph.D. dissertation, UNIVERSITY of CALIFORNIA at BERKELEY, 2002.

[4] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[5] I. Grondman, M. Vaandrager, L. Busoniu, R. Babuska, and E. Schuitema, "Efficient model learning methods for actor–critic control," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 42, no. 3, pp. 591–602, 2012.

[6] B. Goodrich and I. Arel, "Unsupervised neuron selection for mitigating catastrophic forgetting in neural networks," in *Circuits and Systems (MWSCAS), 2014 IEEE 57th International Midwest Symposium on*. IEEE, 2014.

[7] ——, "Neuron clustering for mitigating catastrophic forgetting in feedforward neural networks," in *Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2014 IEEE Symposium on*. IEEE, 2014, pp. 62–68.

[8] R. V. Florian, "Correct equations for the dynamics of the cart-pole system," *Center for Cognitive and Neural Studies (Coneural), Romania*, 2007.