Temporal-difference Learning Analysis - $TD(\lambda)$

Brad Smith

I. Introduction

In 1988, Richard Sutton released a paper called "Learning to Predict by the Methods of Temporal Difference" in which he formalized temporal-difference (TD) learning methods for multistep learning problems. At the time, conventional learning methods revolved around making prediction updates from actual error; his goal was to relate successive predictions and incrementally make prediction updates *during* a process instead of waiting until the result is known at the end. Sutton further provided evidence to show that temporal-difference learning for multistep processes is more accurate and efficient than the more commonly applied supervised-learning methods at the time the paper was written. This paper investigates Sutton's study on the bounded random walk dynamic problem in detail and is successful in backing up his claims. An assumption is made that claim made in this report regarding temporal-difference learning as a better procedure to supervised-learning is regarding multi-step processes.

II. Mathematical Background

A mathematical overview of Sutton's work begins with defining a *learning procedure* to be a set of rules for updating a weight vector, *w*. The goal of a learning procedure is to algorithmically update this weight vector such that error between predicted and actual outcomes is minimized. Sutton defined the weight update rule for a general learning procedure according to (1) for a sequence of *t* observations in a sequence. A *sequence* is a set of time ordered observations where the label, or actual outcome is made known after the sequence. The definition of the change is dependent on the specific learning technique being used [1].

$$w \leftarrow w + \sum_{t=1}^{m} \Delta w_t \qquad (1)$$

In supervised-learning, the weight update rule for a multistep process is defined according to (2). In the context of this paper, α is a learning rate such that $0 \leq \alpha \leq 1$. The learning rate is thought of as a measure of how much "trust" can be placed in the weight update. A larger value would converge faster but potentially not to an optimal value or diverge. A smaller learning rate might lead to trivial convergence if the tolerance criteria is not balanced well, or cause convergence to be relatively slow. Although learning rate is 0 inclusive, a learning rate of 0 implies that no learning is going on and the weights will not change from their initial values. Learning rates can be a function of *t*, but it was assumed constant within each learning procedure for Sutton's experiments.

$$\Delta w_t = \alpha(z - P_t)\nabla_w P_t \qquad (2)$$

In (2), P_t is the prediction of a given vector of observed features x_t at a time, *t*. The quantity $(z - P_t)$ defines the error of a given prediction during the time series to the actual outcome of the sequence (z). The gradient of the prediction at each time is taken so that each error is scaled proportionally to the extent that the given prediction was dependent on a particular weight; this leads to a phenomenon more commonly known as *gradient descent*. An inefficiency in the representation of (2) is that the actual outcome of the sequence, z, is not known until the end, so all predictions must be stored in memory until the end of a sequence to compute the weight update [1].

In "Learning to Predict by the Methods of Temporal Differences", Richard Sutton pointed out an interesting feature of (2) that he exploited to open a new representation of the weight update procedure for supervised-learning. Sutton redefined $(z - P_t)$ according to (3), where $P_{m+1} = z$. In fact, this summation can be expanded out to prove that it is equivalent to the left-hand side. Sutton then showed that substituting (3) into (2), equation (4) is established (using some additional characteristics of summations) [1].

$$z - P_t = \sum_{k=t}^{m} (P_{k+1} - P_k) \qquad (3)$$

$$\Delta w_t = \alpha(P_{t+1} - P_t) \sum_{k=1}^{t} \lambda^{t-k} \nabla_w P_k \qquad (4)$$

Equation (4) provided a method to incrementally update weight vector values without needing to know the result (actual outcome z). The formulation stated that this weight change is dependent on all previous predictions up to the last time step, *t*. Sutton also declared a parameter λ (such that $0 \leq \lambda \leq 1$) to decay the influence of older predictions on newer predictions' weight updates in a consistent manner, with more aggressive trimming taking place at lower values.

To consider the edge cases, when $\lambda = 1$, all previously experienced estimates are accounted for in the weight update computation, which is *mechanically equivalent* to the effect of standard supervised-learning according to (2), through the errors may still be different depending on when the weight updates are applied. On the other extreme, when $\lambda = 0$, the algorithm only considers the previous time for weight updates and "ignores" the older time histories completely. Sutton

defined (4) as the $TD(\lambda)$ learning procedure update equation, an intentional formula that parametrically bridges (using λ) temporal-difference learning with supervised-learning.

By incrementally updating the weights at each time, Δw_t is modified in place, requiring no additional storage of data. This yields the first significant point that Sutton made in his TD research: given a sequence of M steps, because of this incremental weight update ability, 1/M the storage space is used during runtime in comparison to the supervised-learning procedures that were popular at the time, therefor it is proven that TD methods are more *memory efficient*.

The remainder of this report will focus on replication of Sutton's 1988 experiments which leveraged (4) to prove that, for multistep processes, temporal-difference learning using $TD(\lambda)$ is more *accurate* that supervised-learning.

III. Methods

In effort to clearly illustrate the accuracy differences between temporal-difference learning and supervised-learning, replication of Sutton's bounded random walk study was performed. At the time of Sutton's paper, temporal-difference methods had been used in complicated domains before, but by focusing on a simpler dynamical system, elements of accuracy that elevate it about supervised-learning are easier to focus on and should be present [1].

A. Bounded Random Markov Process

The random walk Markov Process is a multi-step process (Figure 1). The agent starts at state D. Then, the agent continuously moves left or right at a probability of 50% into a directly neighboring state until state A or G is reached, and the walk is terminated. If the agent terminates on the left-hand side in state A, the label is 0, whereas if the agent terminates on the right-hand side the label is 1. Each state is represented as a unit basis vector where the state location is the element location. For example, state D is represented as [0, 0, 1, 0, 0]. A sequence is a list of states with a scalar label at the end corresponding to the side of termination.

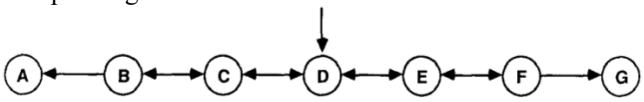

Figure 1: Bounded Random Walk Markov Process [1].

This simple formulation by Sutton was defined so that the weight vector would yield the expected value of z given the current state: 1/6, 1/3, 1/2, 2/3, 5/6 for B, C, D, E, and F, respectively. This was the truth comparison when looking at the RMS error of generated weights for Sutton's experiments. With the weights defined as the expected value of the current state and the unit basis vector convention for defining states, the prediction for a given time is defined as the inner (dot) product of the weight vector and the state vector. Additionally, the gradient is the state vector since each prediction is linearly dependent only on the current state as a function of the weights, where each expectation weight is aligned with each specific state. A new formulation is represented in (5) and was the version of the weight update implemented for this study.

$$\Delta w_t = \alpha(w^T x_{t+1} - w^T x_t) \sum_{k=1}^{t} \lambda^{t-k} x_k \quad (5)$$

When $\lambda = 1$, (5) is mechanically equivalent to the Widrow-Hoff classical supervised-learning rule. The Widrow-Hoff case is highlighted in experiment figures to visual cue the comparison between temporal-difference learning and supervised-learning.

B. Implementation – Python

The experiments were performed using object-oriented Python. With program efficiency being a priority, the NumPy library was used to make available its vectorization capabilities to speed up experimentation. A main program was written to contain a class for the TD learning procedure that was written general enough to be used when running different experiments, using flags to trigger different code paths depending on the study of interest. This was found to be a beneficial experimental method for understanding the details of how Sutton modified the learning method between his experiments.

A python script was developed to generate a predetermined number of training data sets, each with a specific set number of sequences. All sequences were written into a file as an *(n+1)-by-5* matrix for a sequence of *n* non-terminal steps, where the *n+1* row was a single row vector of the label (e.g. [1 1 1 1 1] if result was right hand side termination). This matrix description provided a simple method for reading the sequences into a NumPy array.

The training data creation script was established to give optional intentional favoring of either left-side or right-side terminating sequences to evaluate the extent to which having a set of training sets would be affected if there wasn't enough statistical soundness in Sutton's initial study. Persisting training sets was employed for efficiency in testing and debugging the TD procedure.

IV. Experiments

Sutton ran two experiments using the bounded random walk problem; an effort to re-create them was performed to investigate his arguments in favor of using learning for multi-step processes. For both experiments the same training data sets were used: 100 training data sets, each consisting of 10 bounded random walk sequences. For each dataset, an additional criterion was established to force the distribution within each dataset to be 50% for left and right terminating random walks (5 walks terminating at each side). Sutton does not explicitly explain his training data sets, other than that he chose 100 sets to for statistical reliability. The extent to which 100 sets is statistically reliable and the need to have restrictions on training set termination distributions was examined in further depth as a precursor to replicating both experiments.

A. Experiment 1

The first experiment was replicated to examine the effects on accuracy of the $TD(\lambda)$ when varying λ and averaging converged weight values. The weights were updated according to (5) for each λ, but not applied until after a training data was presented to the learning algorithm [1]. The intent here was to show that, even without updating the weights between sequences temporally, $TD(\lambda)$ can yield more accurate results than supervised-learning methods. For experiment 1, termination of weight convergence was determined once the RMS change in weights became less than 0.01.

B. Experiment 2

The second experiment examined how the value of the learning rate (α from Equation 3) can affect the accuracy of the algorithm when inter-sequence weight updates are made. The focus in this experiment was also to show the benefits of updating the weights between sequences. Error was examined when the learning rate was modified for cases where the training set was presented just one time (rather than until convergence of the weight vector as in experiment 1). The same data was presented as in experiment 1, but weight updates were performed after each sequence, unlike in experiment 1 where they were accumulated and added onto the total weight vector after all sequences were presented to the learning algorithm. Best learning rate values were estimated for each λ, and averaged over 100 training data sets to show more exhaustive results in a second figure. For experiment 2, the weight vector was initialized to 0.5 for all values to prevent right-side or left-side terminations, as Sutton did [1].

V. Results

A. Training Data Distribution Study

During the process of replicating Sutton's 1988 results, an investigation was performed into the effects of weighting the distribution of left and right terminations with a given training data set. When not restricting the distribution of walks within datasets, left, and right terminating cases could be uneven within a given dataset (potentially 4 left ending walks and 6 right ending walks). It was noticed that when this was allowed to happen, there could be large swings in results depending on how many datasets were being trained (10, 30, 50, and so on).

Increasing the number of sequences per dataset to 20 appeared to lower overall error of random walk expectations from the expected weights, but continued to show different results than that of Sutton's experiments. By eventually enforcing a strict equivalency in left and right terminating walks in each dataset, more consistent experiments were achieved in comparison to Sutton's work; this was used for generating the dataset that was created for the experiments to provide more secure statistical reliability.

B. Experiment 1

Figure 2 shows the results from running experiment 1:

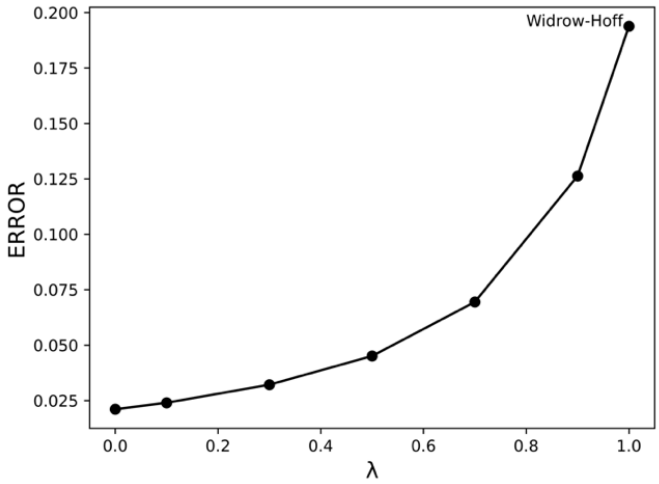

Figure 2: Average error for the bounded random walk under repeated presentations of each dataset until convergence of the weight vector (using RMS change). All initial weights were initialized to 0.5. Convergence occurred within ~5 iterations and $\alpha = 0.1$ for all λ.

This experiment was intended to prove that the algorithm for weight updates given by (5) would converge in finite time, when weight updates are provided at the end of the presentation of a particular sequence to the learning algorithm.

On average, each dataset converged in about 5 iterations for a given value of λ when all weights were initialized to 0.5. It was noticed that as weights were increased or decreased from the average expected value of 0.5, convergence took place on average in closer to 9-15 iterations. Figure 3 shows an example for initialized weight values of 0.0. The weight vectors still converged with errors around the same as in Figure 2. Although the same error distribution was not precise, convergence still occurred, thus validating Sutton's *repeating presentations* training paradigm that the weight vector always converged no matter what the initial weights were set to, just at the expense of more computational time for iterations.

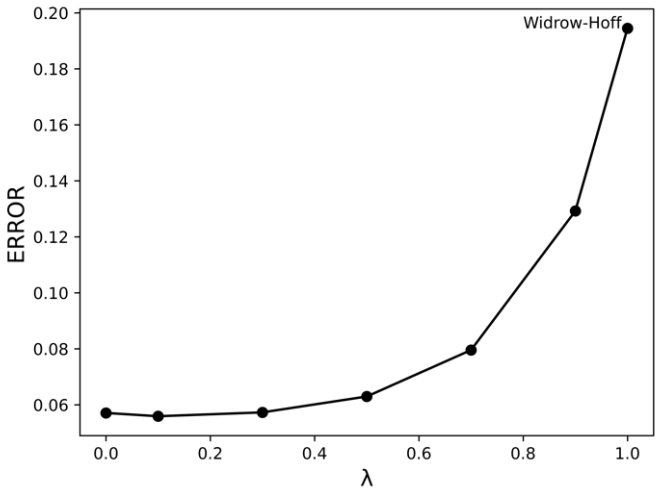

Figure 3: Evidence of convergence when initialized weight vector values are all 0.0. Convergence occurred over a more dynamic range, depending on the value of λ.

Beyond convergence, the replication in Figures 2 and 3 back up Sutton's claim that, although Widrow-Hoff minimizes error on a *training set*, it did not minimize error for future expected values that is made available when discounting previous experience when λ is less than 1 [1]. Initially, it was unclear the learning rate schedule that Sutton used for each λ value, but through experimenting with different values, it was found that consistent qualitative results with respect to Sutton's first experiment were found when using a consistent value across all λ's, where a learning rate of 0.1 showed errors with the same magnitude.

There were a few difficulties in initially replicating these results. The first was related to weighting increment placement. It was learned that by not applying this in the correction location, weights could grow large for certain values of α. Making an initial misstep regarding weight application was one of the greatest challenges of experiment 1.

It was also discovered through experiment 1 the balance of how to determine that the weights have converged enough. The following are examples of weight convergence criteria's that were examined during this experiment:

1. Minimum error weight < error tolerance
2. Maximum error weight < error tolerance
3. RMS error < error tolerance
4. Summation of all weight errors < tolerance

With Sutton not explicitly stating convergence criteria explicitly, this was left as a study. Through testing, option (1) was found to be inadequate as a method since it is not restrictive enough. Options (2), (3), and (4) provided similar results, and ultimately (3) was used to maintain consistency with how error was defined throughout the rest of the text.

Another area of focus was determining reasonable learning rates to schedule as a function of λ. Provided that the intent of experiment 1 was to focus on accuracy of converged rates, maintaining consistent learning rates across all λ maintained similar comparisons between the traded λ values. It is not known what Sutton's choice for learning rate was, although some testing with different rates maintains the same shape of the curve with different errors (until the rate grows to a region of divergence). Therefore, it is reasonable that a difference in chosen learning rates is a primary contributor to the difference in error values between Sutton's first experiment and that represented Figure 1.

C. *Experiment 2*

The second experiment extended experiment 2 to show the benefits of temporal-difference learning when updates to the weight vector are made within sequences (and not just at the end as was shown in experiment 1). Figure 3 clearly shows that when temporal-difference learning is enabled for weight updates state to state, λ less than 1.0 is preferred for reasonable values of learning rates (α). This is the crux of Sutton's argument that has been replicated: for a multi-process, temporal-difference learning will achieve more accurate predictions than supervised-learning (Widrow-Hoff) for any reasonably defined learning rate.

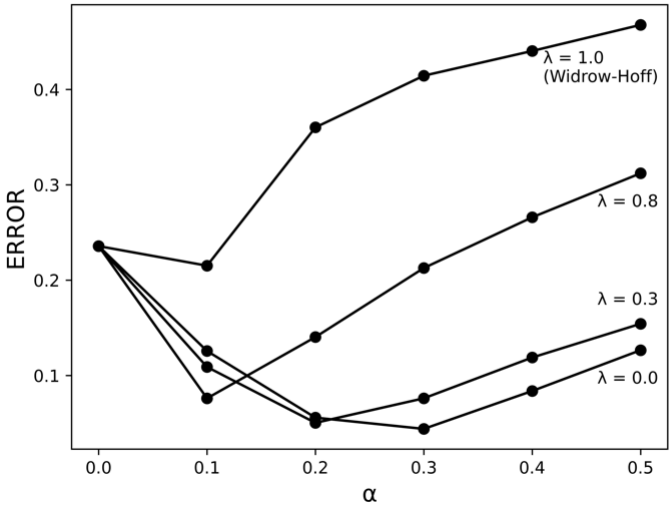

Figure 4: Average error for random walk after experiencing 10 sequences using RMS error to the expected weight values. For $\alpha = 0.0$, no learning has taken place, so this is simply the RMS error between the initial weight vector (all initialized to 0.5) and the expected values.

Taking the results from Figure 4 for best α values for a more values of λ, an average over the entire 100 training data sets was taken to generate average RMS error and recorded in Figure 5. This was intended to show when parameterizing α, an intermediate λ value between 0 and 1 can be determined to minimize error for a given multi-step process. This illustrates the bold point that Sutton was aiming for with his paper. Temporal-difference, when parameters are tuned, should provide globally better accuracy than supervised-learning (shown in Figure 4). The passing of relative prediction errors between sequences for a process with multiple steps has

advantages over comparing each prediction to the actual outcome as in supervised-learning.

For the replicated study, and optimal value of around $\lambda = 0.3$ supports Sutton's reasoning for why $\lambda = 0$ is not the optimal value for this training paradigm: $TD(0)$ is slow in propagating predictions along the sequence [1]. The repeated presentation paradigm allows time for this to occur over multiple iterations which is why the results show $TD(0)$ as optimal for experiment 1, but not experiment 2.

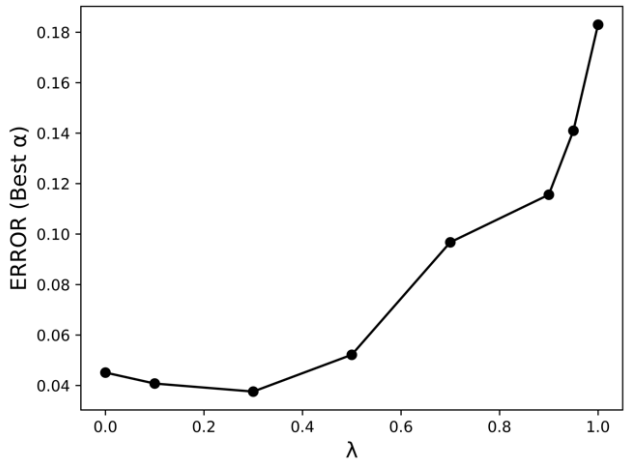

Figure 5: Average error at approximately best α value for random walk using sequence weight update. Each datapoint represents the average RMS error to the expected weight values over 100 training data sets.

The major difficulty in replicating experiment 2 was matching the exact graphs that Sutton prepared in his 1988 paper. It wasn't initially clear how the weight updates were being initialized sequence to sequence, and huge divergence was being seen unless using d small learning rate values on the order of 10^{-2}. It was then noted that the key detail was to make sure that the delta weight vector was re-initialized to 0 at the start of each sequence to ensure that weight additions didn't get counted in multiple times. This mistake was cleared up and results became more closely in line with the qualitative results from Sutton.

D. Combined Experiment Remarks

Although strict numerical dependent variables between Sutton's experiment 2 results and those shown in this report are different, the important detail is the fundamental trend. Figure 4 backs up Sutton's claim that temporal-difference learning will yield more accurate results than supervised-learning for any reasonable learning rate. Then, expanding this to modified values of learning rates for the same training paradigm of incremental weight vector updates within a sequence, Figure 5 states that using temporal-difference learning methods for single iterations through a training dataset (when averaged for many training data sets) yields errors within a reasonable percentage (18% for at maximum for when λ is 1.0).

The result of Figure 5 displays an additional efficiency in temporal-difference learning: a single iteration of the training method on each dataset might yielded more accurate prediction for the weights than the repeated presentation case of experiment 1, in which at best case it was found that weight vector convergence occurred in around 5 iterations for the same initial weight vector values as in experiment 2. Even though in a different problem, convergence using experiment 2 methods would likely be used to allow multiple passes over the training data, the principle exhibited by the single presentation experiment shows the benefit of learning through successive predictions.

As a general commentary, replication of Sutton's results proved that when attempting to optimize a learning procedure for a given process, tuning of parameters like λ and α can be an extensive and challenging step. For a given process with which learning is desired, results can vary intensely with different learning rates, and particular knowledge of the specific problem and orders of magnitude are important for selection of values that provide reasonable results. In this set of experiment, it was found that in cases with larger values of α (above 0.5), divergence could happen rapidly, especially with high values of λ, this was attributed to the phenomenon that with long random walks, the summation in (5) can grow excessively large and thus lead to large shifts in the weight vector without convergence. Convergence in this implementation was capped with an iteration limit, which was used to tune learning rates away from hitting this limit while performing the experiments. Analysis of the preceding nature should be expected when applying TD learning.

VI. Conclusion

At the time that Sutton released "Learning to Predict by the Methods of Temporal-difference", supervised-learning was the preferred learning method. By formalizing temporal-difference learning as a contender for more optimal results for multi-step processes, he was making a bold challenge for the time. Sutton's expression of memory efficiency through his mathematical formulation of $TD(\lambda)$ was verified to show that incremental weights can allow this method to use 1/M the amount of memory for a given sequence with M steps. The difficulties of tuning the training problem were also realized and the knowledge of how to properly tune learning method parameters given and understanding of the problem was another key area of learning during this process. By following Sutton's bounded random walk experiments, convergence of the weight vector was proven, optimally of incremental learning from experience over episode learning was exhibited. This replication and analysis support Sutton's 1988 claim that temporal-difference learning is more accurate and efficient that supervised-learning methods for multi-step processes.

References

[1] Sutton, R.S. *Learning to Predict by the Methods of Temporal-differences*. Mach Learn 3, 9-44 (1988). https://doi.org/10.1007/BF00115009