Replication of Sutton, 1988

[*]Hash for latest commit to the GitHub repository - 585e3c40344165bb20df8d1b6c6a6f43a7db6a72

Prithi Bhaskar

pbhaskar8@gatech.edu

***Abstract*—This article highlights the advantages of incremental learning procedures specialized for prediction. This is done by implementing and replicating the results from Sutton(1988).**

***Index Terms*—Incremental Learning, TD(λ), Supervised Learning**

I. Introduction

Sutton(1988) introduces and provides the first formal results in the theory of temporal-difference TD methods, a class of incremental learning procedures specialized for prediction problems. Whereas conventional prediction-learning methods are driven by the error between predicted and actual outcomes. TD methods are similarly driven by the error or difference between temporally successive predictions; with them, learning occurs whenever there is a change in prediction over time. It also shown that TD methods have two kinds of advantages over conventional prediction-learning methods. First, they are more incremental and therefore easier to compute. The second is that they tend to make more efficient use of their experience: they converge faster and produce better predictions. This article replicates the results provided in Sutton(1988) that are presented as proofs for the above claims.

II. Approaches to prediction

A. Prediction Problems

Multi-step prediction problems are those in which experience comes in observation-outcome sequences of the form $x_1, x_2, x_3, ..., x_m, z$, where each xt is a vector of observations available at time t in the sequence, and z is the outcome of the sequence. Many such sequences will normally be experienced. For each observation-outcome sequence, the learner produces a corresponding sequence of predictions $P_1, P_2, P_3...., P_m$, each of which is an estimate of z. The predictions are also based on a vector of modifiable parameters or weights, w. All learning procedures will be expressed as rules for updating w. Mathematically, it is written as

$$w = w + \sum_{t=1}^{m} \Delta w_t \tag{1}$$

B. Supervised-learning

In the conventional supervised-learning setting, the learner is asked to associate pairs of items. When later presented with just the first item of a pair, the learner is supposed to recall the second. Any prediction problem can be cast in the supervised-learning paradigm by taking the first item to be the data based on which a prediction must be made, and the second item to be the actual outcome, what the prediction should have been $[Sut88]$. In Sutton's paper and this article as well, this approach to a prediction problem is referred to as the supervised-learning approach. In the supervised learning setting, the increment in weight in (1) is calculated as a function of the error between P_t and z and change in P_t due to change in w. This can be written as:

$$\Delta w_t \ = \alpha(z - P_t)\nabla_w P_t \tag{2}$$

If the prediction is just a linear function of x_t and w, i.e. P_t= $w^T x_t$, (2) reduces to the below equation which is also called the Widrow-Hoff rule(Widrow Hoff. 1960):

$$\Delta w_t = \alpha(z - w^T x_t)x_t \tag{3}$$

C. TD learning

TD procedure produces exactly the same result as (2), and yet it can be computed incrementally. The key is to represent the error z - P_t as a sum of changes in predictions $[Sut88]$. This can be mathematically expressed as below:

$$\Delta w_t = \alpha(P_{t+1} - P_t)\sum_{k=1}^{t} \nabla w P_k \tag{4}$$

Sutton's claim is that (4) makes much milder demands on the computational speed of the device that implements it and that it also saves substantially on memory, because it is no longer necessary to individually remember all past values of $\nabla w P_t$.

D. TD(λ)

TD(λ) is a class of procedures that make greater alterations to more recent predictions, λ here refers to an eligibility trace. In TD(λ), alterations to the predictions of observation vectors occurring k steps in the past are weighted according to λ^k for $0 \leq \lambda \leq 1$. This can be thought of as similar to the n-step TD methods but with some computational advantages. The primary advantage is that, only a single trace vector is required to be stored rather than a store of the last n feature vectors(the $\nabla w P_k$ component of (4)). Also, while the n-step TD methods can be looked at as forward views, the TD(λ) methods can be looked at as backward views as it looks backward to recently

visited states and updates them based on the current TD error. The update rule for TD(λ) is given by:

$$\Delta w_t = \alpha(P_{t+1} - P_t) \sum_{k=1}^{t} \lambda^{t-k} \nabla w P_k \tag{5}$$

For λ =1, (5) is equivalent to (4), the TD implementation of the prototypical supervised-learning method and the procedure given by (4) is referred as TD(1). For $\lambda = 0$, the weight increment is determined only by its effect on the prediction associated with the most recent observation and the equation reduces to:

$$\Delta w_t = \alpha(P_{t+1} - P_t) \nabla w P_t \tag{6}$$

TD methods (using (5))make more efficient use of their experience than do supervised-learning methods and converge more rapidly and make more accurate predictions. TD methods have this advantage especially when the data sequences have a certain statistical structure that is ubiquitous in prediction problems. This structure naturally arises whenever the data sequences are generated by a dynamical system, that is, by a system that has a state which evolves and is partially revealed over time. The below section describes and implements one such system where TD methods are more efficient than conventional supervised-learning methods.

III. Random Walk Experiment

Sutton 1988 uses a random-walk example to demonstrate the advantages of TD methods over supervised-learning methods. This article uses the same and replicates the experiments detailed in Sutton 1988 and in later sections analyses the similarities and differences between the two. Figure 1 shows a model that generates a bounded random walk - a state sequence generated by taking random steps to the right or to the left until a boundary is reached. Every walk begins in the center state D. At each step the walk moves to a neighboring state, either to the right or to the left with equal probability. If either edge state (A or G) is entered, the walk terminates. The objective of the experiment is to estimate the probabilities of a walk ending in the rightmost state, G, given that it is in each of the other states.

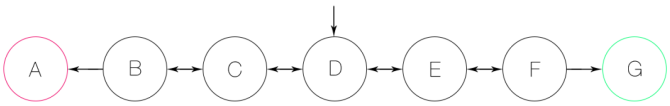

Figure 1. Random walk Example

IV. Implementation

Since the objective is to estimate the probabilities of a right-side termination, a walk's outcome was defined to be z = 0 for a walk ending on the left at A and z = 1 for a walk ending on the right at G. The learning methods were then provided with training sequences and made to estimate the expected value of z which is equal to the probability of a right-side termination.

A. Generation and Representation of Sequences

As mentioned in the original paper, every walk was assumed to start from state D. From D, the subsequent states were generated from an uniform distribution over the immediate left and right states until either of the terminal states is reached. If the walk terminated in state A, the outcome of the sequence was assigned as 0 and if the walk terminated in state G, the outcome was assigned as 1. So a sequence would be of the form $X_d, X_c, X_b, X_a, 1$. Furthermore, the states were represented as unit basis vectors of length 5, that is, four of their components were 0 and the fifth was 1, with the one appearing at, a different component for each state. For e.g., state D takes the form $(0, 0, 0, 1, 0, 0, 0)^T$.

B. Training Sets

For all the three experiments described in Sutton 1988, 100 training sets each consisting of 10 sequences were used. Similarly, 100 training sets were generated from the model described in the previous section and each of the sets consisted of 10 random walk sequences. The training sets were then presented either repeatedly or only once to the learning algorithms depending on the experiment.

C. Data

Due to the usage of just 100 training sets(compared to the infinitely possible sequences), the number of left-terminating sequences and right-terminating sequences generated were identified for every 25 sets and plotted. This was done to verify that no particular outcome ends up being under-represented or over-represented. Since the sampling of next states is done randomly, the number of sequences ending at the rightmost state G is always almost equal to the number of sequences ending at the leftmost state A. Figure 2 shows the distribution of sequences ending in right and those ending in left across the training set.

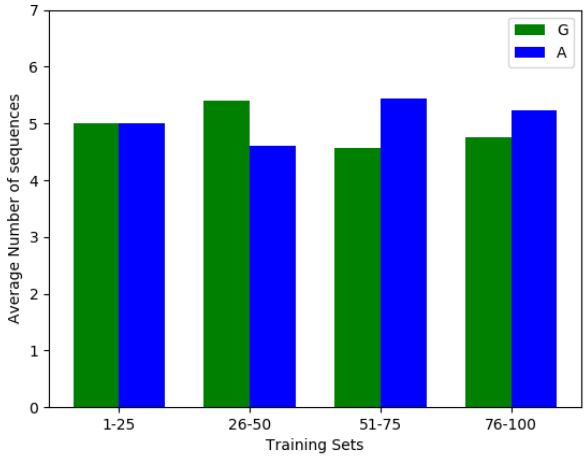

Figure 2. Distribution of Data in the training sets

As can be seen in Figure 2, on an average, both the right-terminating sequences and left-terminating sequences are al-

ways equally represented because of the usage of uniform distribution in selecting future states.

D. TD(λ)

The TD(λ) learning procedure given by (5) was implemented to take different values of lambda, alpha and weight vector of length 5(one for each of the non terminal states). It returns the value Δw_t for each sequence. As mentioned earlier, substituting λ =1 in (5) is equivalent to the prototypical supervised-learning method and hence the results of various values of λ ranging from 0 to 1 were used to compare the performances of incremental learning and supervised learning approaches.

Also, since the states are represented as unit basis vectors, if the state the walk was in at time t has its 1 at the i^{th}component of the observation sequence, then the prediction $P_t = w^T x_t$ is simply the value of the i^{th} component of w. This is found by simply calculating the dot product of the weight vector at time t and the input sequence till time t. Hence the incremental weight update is calculated by doing a product of alpha, the difference in successive predictions and the λ sequence. Once a terminal state is encountered, the prediction of that state is simply the outcome z of the sequence. The λ sequence is generated by multiplying the already available sequence by λ and then appending a new λ at the end for the most recent observation. This is done at the end of all the steps in a sequence.

V. Replication

A. Experiment 1

As described in Sutton 1988, experiment 1 calculates the probability of a right-side termination for each of the non terminal states for different values of λ. As the learning procedures are provided with more and more sequences, the calculated probabilities are expected to converge to the true probabilities of each of the non terminal states. The true values for states B, C, D, E and F are calculated as $1/6, 1/3, 1/2, 2/3$ and 5/6 respectively (Section 4.1., Sutton, 1988). The root mean square (RMS) error between the predictions from the learning procedures and the true values were then used as a measure of the performance of the learning procedures. As mentioned in the original paper, the learning procedures were presented with each of the training sequences repeatedly until convergence (Convergence threshold for generating Figure 3 was 1e-6). This convergence criteria was determined empirically and the procedure for the same has been explained in Section VI.B.

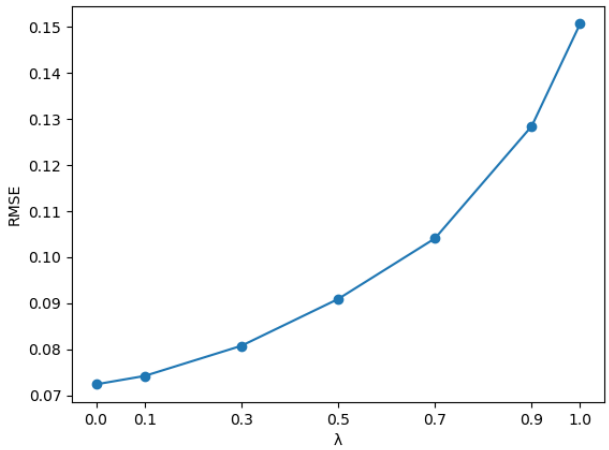

Figure 3. Replication of Figure 3 of Sutton, 1988

As mentioned in the paper, the weight updates were done only at the end of a complete training set and not at the end of each of the sequences. The Δw values were accumulated over a complete training set(10 sequences) and then added to the weight vector after the complete presentation of the training set. Also, the value of alpha used for generating the plot shown in Figure 2 was 0.001. This was chosen empirically by trying a lot of values and the value that caused the weight vectors to always converge consistently was chosen as the final value.

As shown in Figure 3, the RMSE is the lowest for TD(0) and is the highest for Td(1), the Widrow-Hoff procedure. The error values plotted are the RMSE values averaged over all the training sets.

1) Comparison of results: Even though the values shown in Figure 3 are not the exact same values as in the original paper, the direction of increase of the error is the same(TD(1) always ends up with a higher error than TD using intermediate λ values) each time the experiment is run, asserting the efficiency of incremental learning procedures. This difference in the values might be because of the difference in the nature of the training sets. Also, the values of errors for each λ value differ between each run depending on the randomly generated training set. However, as mentioned in the original paper, for very small values of alpha the weights always converged to the same values for a particular training set, irrespective of their initial values. Also, since the training sets are presented to the procedures over and over again until convergence, the learning procedures are at their best, as can be ascertained from the very low RMSE values.

Another difference that can be observed is that the curve for λ values less than 0.5 is more flat in the original paper when compared to Figure 3. This might be because of the difference in the convergence criteria used in the experiments. As the convergence criteria involves lower and stricter thresholds, the difference in the learning procedures might seem more pronounced.

B. Experiment 2

The second experiment concerns the question of learning rate when the training set is presented just once rather than repeatedly until convergence. The same data used in experiment 1 was presented to the learning procedures again, for several values of λ along with the following procedural changes as mentioned in the original paper: First, each training set was presented only once to each procedure rather than repeatedly until convergence. Second, weight updates were performed after each sequence rather than after each complete training set. Third, each learning procedure was applied with a range of values for the learning rate parameter. Fourth, so that there was no bias either toward right-side or left-side terminations all components of the weight vector were initially set to 0.5. Similar to experiment 1, RMSE values were plotted against the various alpha values and for various learning procedures. The results are shown in Figure 4. As in experiment 1, the values shown in the plot are RMSE values averaged over all the training sets.

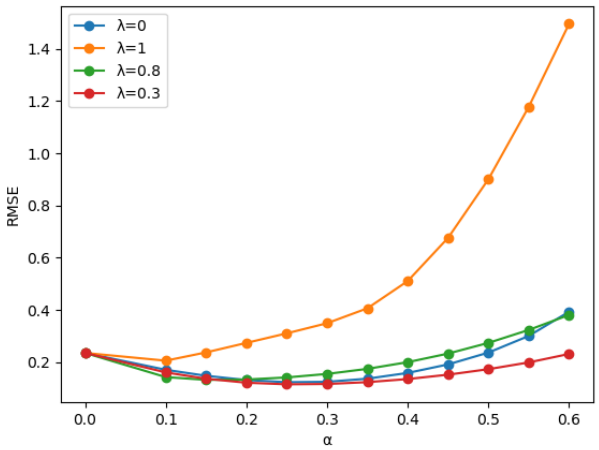

Figure 4. Replication of Figure 4 of Sutton, 1988

1) Comparison of results: As in the case of the first experiment, the RMSE values in Figure 4 are not the same as those presented in the original paper but the learning curves of the various learning procedures are similar to the ones presented in the paper. All of the TD methods with λ less than 1 performed better over a wider range of α values than the supervised-learning method. This proves that TD methods make more efficient use of their experience and makes better predictions when compared to the conventional supervised learning methods. In particular, the errors for various λ values were the lowest at α=0.3 and also λ value of 0.3 outperforms all the other values for almost all α values. This trend is just the same as the one presented in the original paper. However, in the λ=1 case, the RMSE value was much higher than the TD(λ) procedures than the ones shown in Figure 4.

All λ values below 1 perform much better than the supervised learning procedure and as all the parameters for the experiment were the same as discussed in the original paper(the learning rates and lambda values, the number of times the training set was presented, the method of calculation of the mean RMSE), the only parameter that can explain the difference between the graph presented in the original paper and Figure 4 is the training data used. In the training data used for replication purposes, the difference is not as pronounced as in the original paper. All the learning curves start at the same point as shown in the original paper, which is expected, as they all begin with the same weight vector and no change in the weight vector happens for the case of α=0.

Since the procedures were presented with the training sets only once, as compared to the previous experiment, and also because of the comparatively higher learning rates, the mean RMSE values are higher than the ones observed in experiment 1.

C. Figure 5 of Sutton,1988

Figure 5 of the original paper plots the best error level achieved for each λ value, that is, using the α value that was the best for that λ value. It is assumed that Figure 5 was generated using the same procedures as in experiment 2 : each training set was presented only once to each procedure rather than repeatedly until convergence and weight updates were performed after each sequence rather than after each complete training set. Using the data generated for the above experiments and the above listed procedures, figure 5 of Sutton,1988 was plotted and is shown in Figure 5.

As in the previous cases, the RMSE values shown are the values averaged over the 100 training sets. Except here, the values were averaged for various values of α and then the lowest value among those was plotted for each λ. Since the values of α used to generate Figure 5 were not explicitly listed in the original paper, the range of values used for experiment 2 was reused for generating Figure 5.

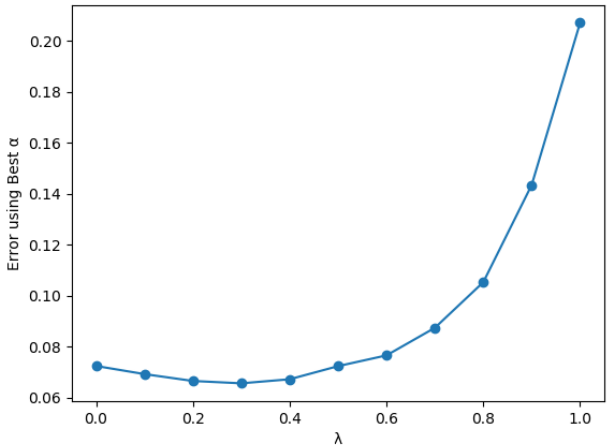

Figure 5. Replication of Figure 5 of Sutton, 1988

1) Comparison of results: As in the above two experiments, all λ values less than 1 were superior to the λ = 1 case.

However, in this experiment as well, similar to the second experiment, the best λ value was not 0, but somewhere near 0.3. This is the same value as shown in the original paper. Overall, the trend of the graph is the same as the one shown in the original paper - There is a decrease in error initially until λ=0.3 and for higher values of λ, the error increases. Since the values plotted are those that are the lowest among different alpha values, the RMSE values are much lower compared to the values in Figure 4. The reason λ = 0 is not optimal for this problem is that TD(0) is relatively slow at propagating prediction levels back along a sequence(Sutton, 1988). As shown in equation(6), with TD(0), only the weights corresponding to the most recent observation are changed and the rest of the weights remain unchanged based on a single observation sequence. Hence the rate of propagation of the approximated-true values to other states is slow. However, this can be overcome by presenting the training sets repeatedly to the procedure and this explains the better performance of TD(0) in experiment 1 compared to experiment 2.

VI. Problems Encountered

A. Implementation of TD(λ)

For the purpose of replicating the above experiments, equation (5) was implemented so that it can be passed with different values of λ for various learning procedures. One of the major challenges encountered while implementing the same was the multiplications involved with matrices. For example, multiplications were not possible when the sizes of matrices were not compatible. The Numpy documentations and examples helped overcome them. Solutions involved adding an additional axis to the existing matrix or an additional dimension so that the matrices become compatible for multiplication.

The second major challenge was figuring out how to calculate predictions P_{t+1} and P_t in the equation (5). The pointers in the original paper helped overcome this issue - If Predictions are just linear functions of input x_t and the weight vector w, then $P_t = w^T x_t$. Hence it was assumed that for this exercise(random walk), prediction at time t can be calculated as the dot product of the weight vector at time t and the input sequence till time t as it is a linear function of the input sequence and the weight vector. And then there is the gradient part of the equation - $\nabla \mathrm{w} P_t$. Again since the predictions are linear functions of input x_t and the weight vector, w, the gradient part $\nabla \mathrm{w} P_t$ becomes just x_t, the input sequence observed till time t.

B. Experiments

Most of the procedures used for plotting the graphs in Sutton, 1988 have been detailed in the paper except for a few. For example, the notion of convergence in experiment 1. However, it has been mentioned in the paper that each training set was presented repeatedly to each learning procedure until the procedure no longer produced any significant changes in the weight vector. For the purpose of this assignment, this threshold was set at 1e-6 i.e. the algorithm was considered to have converged when the difference between weight vectors after successive iterations was less than or equal to 1e-6. This value was obtained empirically after trying different values for the threshold. Higher values of the threshold produced inconsistent results across different runs and the lowest value that produced consistent results across runs and also ended up with the same weight vectors for a particular training data was chosen as the final value.

Another problem was about the procedure of updating the weight vector for experiment 2. It has been mentioned in the original paper that for experiment 2, weight updates were performed after each sequence rather than after each complete training set. However, just updating the weight vector after each sequence did not produce results similar to those presented in the original paper. Hence for experiment 2, it was assumed that the weight vector must be reset to its initial value of 0.5 after each training set was presented. However, this has been mentioned indirectly in the paper - each training set was presented once to each procedure[$Sut88$]. From this statement, it follows that the weight vector should be reset after the presentation of each training set, as opposed to the procedure in experiment 1, where the weight vectors were not reset until convergence.

For figure 5, information regarding the range of alpha values used to plot the figure was not provided in the original paper. However, Since the Figure 5 of Sutton's paper was generated with the same experimental setup as experiment 2, the values used with experiment 2 were reused for replicating Figure 5. As in experiment 2, the weight vectors had to be reset after a single presentation of each training set since the procedure requires that each training set be presented only once to each procedure.

C. Dataset

Another big challenge in replicating the figures was to identify a single training set that produced all the three graphs consistent with the ones presented in the original paper. Even though there were a number of training sets that produced each of the graphs exactly the same as the ones in Sutton(1988), finding a single set of training data that produced all the three graphs consistent with the original paper was a time-consuming procedure. A number of different sequences were generated using python's random seed method and the graphs were plotted for those sequences. The seed value that produced all the three graphs, preserving the trends shown in Sutton(1988) was finalised and the graphs plotted using the sequence generated using that seed value have been presented in this paper.

References

[1] Sutton, R.S. (1988) *Learning to Predict by the Methods of Temporal Differences* . Machine Learning, 3, 9-44. http://dx.doi.org/10.1007/BF00115009

[2] Sutton, R. S., Barto, A. G. (2018). 7.1 n-step TD Prediction *Reinforcement learning: An introduction.*MIT press.

[3] Sutton, R. S., Barto, A. G. (2018). 12.2 TD(λ) *Reinforcement learning: An introduction.*MIT press.

[4] Isbell, C., Littman, M.L. *Reinforcement Learning : TD and Friends* Retrieved from https://classroom.udacity.com/courses/ud600