CS7642 PROJECT 1 REPORT – TD(λ)

Dixon Domfeh
de687ae06f50f72af37aba5c80fb659c773aff98(git hash)

***Abstract*—**This report is about replicating the experiments in Richard Sutton's 1988 seminal paper "*Learning to predict by the methods of temporal differences"*. Specifically, we outline the step-by-step approach taken to replicate the results of the bounded random walk experiment.

I. INTRODUCTION

In Sutton (1988), the author proves the convergence of temporal differences (TD) methods, their optimality and show how it is compares with supervised learning. Sutton posits that TD can supplant supervised learning as an alternate to solve real-world dynamic system problems. Sutton further brings to bear several desirable qualities of TD over supervised learning such as low memory overhead, efficient use of experience, and accurate prediction via mathematical proofs and simulated experiments. One of such simulated games is the bounded random walk. TD methods are a family of incremental learning algorithms for prediction problems. They solve the credit assignment by means of temporal differences between successive predictions. Learning occurs whenever there is a change in prediction over time, making efficient use of experience. This contrasts with supervised learning or outcome-based approaches which utilize observation-outcome pairs ignoring the sequential structure of a dynamic system.

Let us consider an observation-outcome of sequence $x_1, x_2, \dots, x_m, z$ where z (a real-valued scalar) is the outcome, $x_1, x_2, \dots, x_m$ are observations/measurements preceding the outcome. Also consider $P_1, P_2, \dots, P_m$ as the prediction of the measurement and an associated weight vector ($\vec{\omega}$) as the loadings such that the prediction at time (t) is a function of the weights and measurement, $P(x_t, \omega)$. In the supervised learning context, a simple update rule (like the perceptron rule) for the weight, ω can be expressed as

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

which is can be updated after each sequence. The change in weight (delta rule) is simply

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

where $\nabla_w P_t$ is the partial derivative with respect to P_t and a learning rate, α. The TD approach for the delta rule is done incrementally. The error, $z - P_t$ which is represented as the sum of the changes in successive predictions can be represented as

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

Substituting equation (3) in equation (1), the update rule can be simplified as

$$\omega \leftarrow \omega + \sum_{t=1}^{m} \alpha(P_{t+1} - P_t) \sum_{k=1}^{t} \nabla_w P_k \qquad (4)$$

A. *TD(λ) Procedure*

The update procedure in equation (4) is just a special case in which the prediction in decreases (increases) are changed to the same extent (Sutton, 1988). When we consider an TD procedure with an exponential decay rate which lays emphasis on more recent predictions, it leads to a generalization of TD(λ). The prediction of measurements k steps in the past are weighted exponentially as λ^k and the new update rule can be written as

$$\Delta\omega_t = \alpha(P_{t+1} - P_t) \sum_{k=1}^{t} \lambda^{t-k} \nabla_w P_k \qquad (5)$$

where $0 \leq \lambda \leq 1$. The variable under the summation sign is the error at time t.

Interestingly, when $\lambda = 1$ equation (5) becomes the supervised/outcome-based learning in equation (4) which can be referred to as the TD(1) or Monte Carlo rule. With $\lambda < 1$ particularly TD(0) (where $\lambda = 0$), the weight update is determined only by the prediction associated with the most recent observation. The difference in the weights change the most in the case TD(0).

II. RANDOM WALK EXPERIMENT

A. *Setup and Data Generation*

Sutton demonstrated that TD methods can learn more efficiently than the supervised learning (or Widrow-Hoff) approach by setting up an experiment of a simple stochastic system. Figure 1 below is a diagram for the bounded random walk sequence.

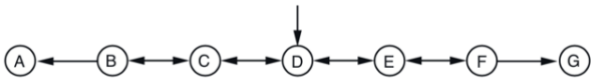


Figure 1: Random walk sequence of state transitions

The sequence consists of seven states with two of them being the absorbing or terminal states (i.e., A and G) and the remaining states (i.e., B, C, D, E and F) are

nonterminal states. When in any state, there is a 50% probability of transitioning to another state (either to the left or right). The walk always begins at state D. The outcome (or reward) of the walk is defined as $z = 0$ when the agent ends up in state A and $z = 1$ when she ends up in state G. For an example if the path followed was DCDEFG then the sequence of observation-outcome would be $x_D, x_C, x_D, x_E, x_F, 1$. This sequence vector (with emphasis on state D's index) can be one-hot encoded as $x_D = (0,0,1,0,0)^T$. A collection of a sequence index vector can be represented as

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

for states x_D, x_C, x_B, x_C, x_F as an example. We randomly generated the 100 training datasets with each consisting of 10 sequences using the following steps:

1. initial state="D"
2. sequence = [states[initial state]]
   reward = None
3. **while** reward is None:
   s = sequence[-1]
   sequence.append(s.left
   if random() < 0.5 else s.right)
4. **if** sequence[next state] == "A":
5. **then** reward = 0
6. **else if** sequence[next state] == "G":
7. **then** reward = 1
8. **end**
9. **return** sequence, reward

To enable replication of our dataset generated we set a random number generating seed which can be found in the Git repository for this assignment.

III. REPLICATION OF RESULTS

This section outlines the detailed steps taken to replicate Figures 3, 4, and 5 of Sutton's paper. We start off by solving the theoretical true value of the weight vector ($\vec{\omega}$). This is needed to measure the error between the TD's prediction and the actual weights.

A. *Calculating the theoretical values of ω*

Remember that the prediction P_t is just a weighted expectation of learnable weights and current state x_t which can be expressed as

$$P_t = \omega^T x_t = \sum_i \omega(i) x_t(i)$$

where i represents the nonterminal states. Let $\boldsymbol{Q}$ be the transition matrix with entry Q_{ij} being the probability of transitioning between nonterminal states i and j. Let $\boldsymbol{h}$ be probability vector between the nonterminal states i and the absorbing state (G). Then it follows from Sutton (1988) that expected true value of outcome, z can be expressed as

$$\mathbb{E}\{z|i\} = \left[\sum_{k=0}^{\infty} \boldsymbol{Q}^k \boldsymbol{h}\right] = [(\boldsymbol{I} - \boldsymbol{Q})^{-1}\boldsymbol{h}]_i \qquad (6)$$

where $\boldsymbol{I}$ is an identity matrix with ones at the diagonal entries. Due to the 50% probability of transitioning, the matrix $\boldsymbol{Q}$ and vector $\boldsymbol{h}$ can be computed for the 5 nonterminal and 1 terminal state (G) respectively as $(\mathbf{5} \times \mathbf{5})$ and $(\mathbf{5} \times \mathbf{1})$ with 0.5 probability each transition state. We solve the dot product of equation (6) for nonterminal state B,C,D,E,F as

$$\left(\boldsymbol{I} - \begin{bmatrix} 0 & 0.5 & 0 & 0 & 0 \\ 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0.5 & 0 & 0.5 & 0 \\ 0 & 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0 & 0.5 & 0 \end{bmatrix}\right)^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.5 \end{bmatrix}$$

to get the true weight vector ω^* which is equivalent to the "*ideal predictions*" $-\frac{1}{6}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}$ *and* $\frac{5}{6}$ -- from pp. 12 of Sutton (1988). The weight ω^* is compared to the TD(λ) estimate of weight to get the prediction error.

B. *Repeated presentation*

In this section we present the result of the random walk experiment under repeated representation. The repeated presentation simply means that the training dataset is being repeatedly presented to the TD (λ) algorithm until convergence of the weight vector ($\vec{\omega}$). The error or RMSE is calculated at the end of each run through the dataset and the final error is averaged over the entire dataset (see Algorithm 1). We replicated Figure 2 as follows. We run Algorithm 1 below with some initial parameters. The convergence threshold is set as $\epsilon = 0.001$ over a maximum iteration of 1000. The learning rate, $\alpha = 0.01$. Initialized weight vector was not explicitly stated in Sutton's paper. In this replication we set an unbiased initial weight $\vec{\omega} = \{0.5, 0.5, 0.5, 0.5, 0.5\}$. We use a range of $\lambda = \{0, 0.1, 0.3, 0.5, 0.7, 0.9, 1\}$. The change in weights, $\Delta\omega$ is accumulated for each run of sequence to obtain the total $\Delta\omega$ which is then added to ω to get the final weight update for each of the training dataset.

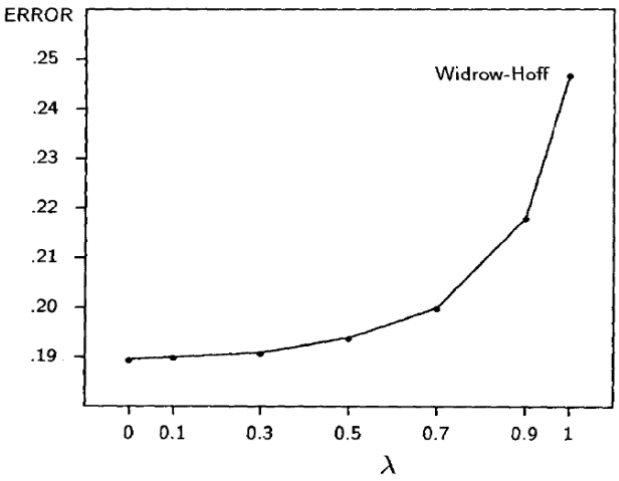
TD(λ) Algorithm 1 under repeated presentation

Input: λ values

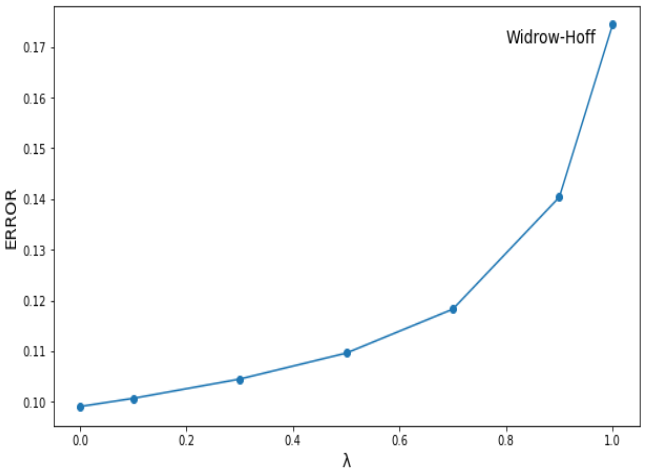
Output: RMSE scores

1. Initialize α, $\vec{\omega}$ and ϵ
2. **for** λ in λ values **do**

for each training set **do**
    for range(max iterations=1000) **do**
        initialize a copy of $\vec{\omega}$ as old_w
        initialize total $\Delta\omega$ to zeros
            for each sequence **do**
                initialize error vector to zeros
                initialize $\Delta\omega$ to zeros
                for each state **do**
                    calculate error
                    calculate $\Delta\omega$
                    sum of $\Delta\omega$
                end
            update weight **(**total $\Delta\omega$, ω)
            end
        test convergence: **(**old_w- ω) $< \epsilon$
        end
        calculate RMSE
    end
    calculate mean RMSE over datasets
end



(a)



(b)

Figure 2: Average error under repeated presentation, (a) is Sutton's[1] and (b) is author's replication.

In general, the two plots in Figure 2 show a similar trend even though there might be some slight discrepancy in the error scores. We reason that this is due to the stochastic nature of the experiment. Factors such as weight initialization, learning rate and randomly generated sequences can be some of the causes of the discrepancy. We observe the worst performance with TD(1). Even though TD(1) may minimize the error between prediction and actual outcome on in-sample data, it does not guarantee a better prediction on out-of-sample observations. TD(0) on the other hand records the lowest error score since it only concerns itself with the most recent observation. Since we run the algorithm repeatedly over the finite dataset many times, we will end up with the maximum likelihood estimate which is the same as TD(0). This is because transitioning from S_{t-1} to S_t happens with some stochasticity. So, for each run over the finite data sample there's going to be a different transition probability. Averaging the transition probabilities over the repeated runs gives the expected value estimate.

C. Single presentation

In the single presentation set up, we run the Algorithm 2 below on the same data set just once, and this time the weight update is done at the end of each sequence. This means that the algorithm learns only 10 sequences of random walk at a time in the training data set. We iterate over different values of α and λ to find the best hyperparameters and check the impact of learning rate out the prediction error. We specifically use the ranges

$\lambda = \{0, 0.1, 0.3, 0.5, 0.7, 0.9, 1\}$ and

[1] We used the corrected plot for Sutton's paper which was found in the Erratum section

$\alpha =$
$\{0.0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.6\}$.

The weight vector initialization is the same as the repeated presentation case. The average error is calculated in the same was as in the repeated presentation set up.

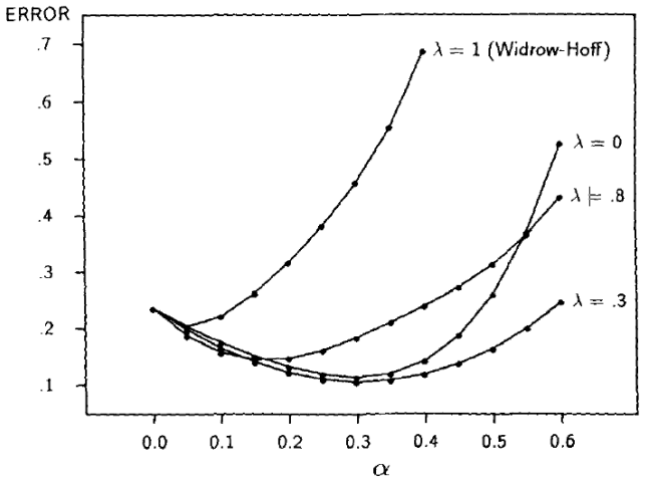
TD(λ) Algorithm 2 under single presentation

Input: λ and α values

Output: RMSE scores

1. Initialize $\vec{\omega}$ and set $\epsilon = 0.001$
2. **for** λ in λ values **do**
3. **for** α in α values **do**
4. **for** each training set **do**
5. **for** range(max iterations=1) **do**
6. initialize a copy of $\vec{\omega}$ as old_w
7. **for** each sequence **do**
8. initialize error vector to zeros
9. initialize $\Delta\omega$ to zeros
10. **for** each state **do**
11. calculate error
12. calculate $\Delta\omega$
13. sum of $\Delta\omega$
14. **end**
15. update weight (total $\Delta\omega$, ω)
16. **end**
17. test convergence:(old_w- ω) $< \epsilon$
18. **end**
19. calculate RMSE
20. **end**
21. calculate mean RMSE over datasets
22. **end**
23. **end**

Figure 3 (a)-(b) below presents the results using the selected $\lambda = \{0,\ 0.3,\ 0.8, 1\}$ combined with different learning rates as seen in Sutton's paper. The learning rates has a significant effect of TD's prediction error. At lower values of α (0.1, 0.2, 0.3), the prediction error is lowest for $\lambda = \{0,\ 0.3,\ 0.8\}$. This explains that optimal weights can be achieved with an α of 0.2. Interestingly, the same range of α values do show an increasing trend in prediction error for the Widrow-Hoff or TD(1) case (red line with $\lambda = 1$) illustrating the limitation of the supervised learning approach. TD($\lambda < 1$) seem to produce lower error with varying learning rates. Our replication results are similar to Figure 4 in Sutton's paper in terms of the error range. We do however observe that the lines corresponding to $\lambda = 0$ and $\lambda = 0.8$ intercept at around an α value of about 0.35 whereas this same intersection occurs an α value of about 0.55 in Sutton's paper. We reason that this slight variation may be due to the random data generation process.



(a)

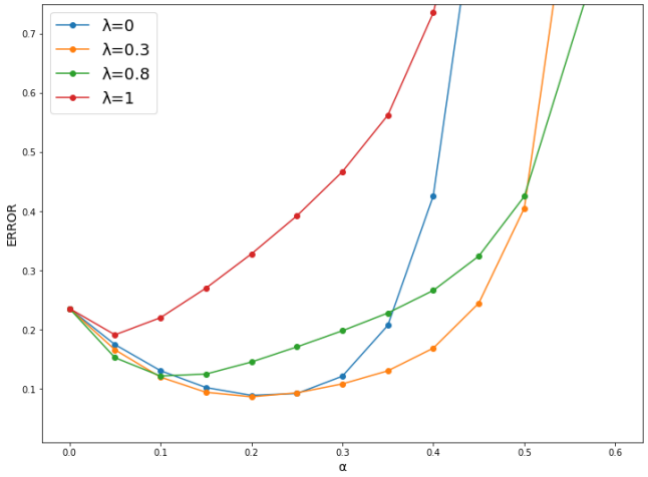
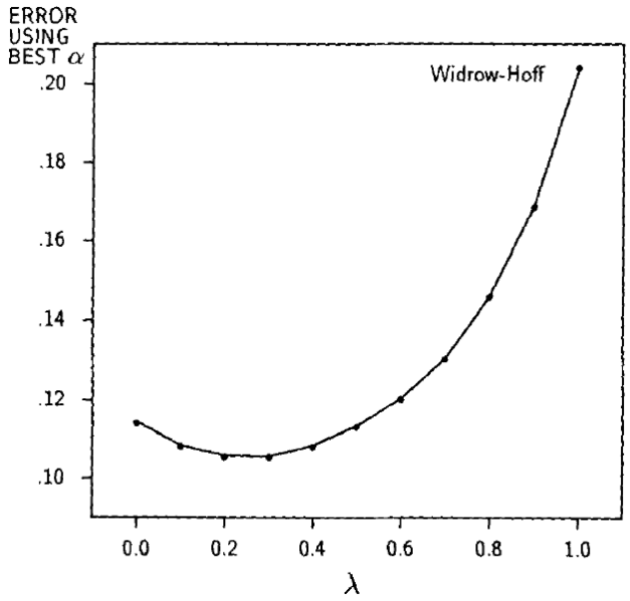
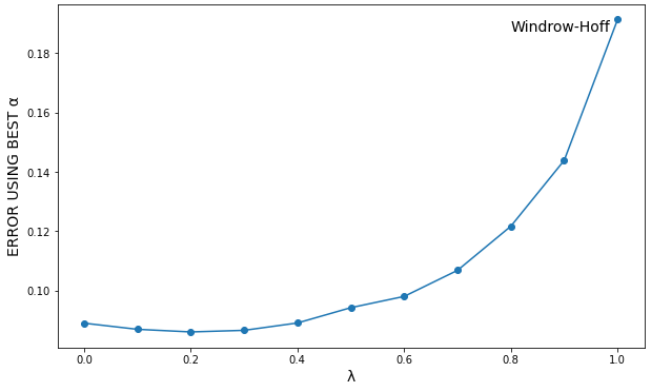


Figure 3: Average error under single presentation: (a) is Sutton's and (b) is author's replication.

Finally, we also select the best α value which corresponds to the lowest root mean squared error across various values of λ. Figure 4 (a)-(b) below compares our result to Sutton's. Our plot shows a trend to Sutton's although not the same. The reason for this discrepancy is the same as previously discussed. Even with the best learning rate the TD(1) or Widrow-Hoff still shows the worst prediction error.



(a)



(b)

Figure 4: Average error using the best α value under single presentation: (a) is Sutton's and (b) is author's replication.

IV. CONCLUSIONS

Thus far, this report has outlined the methodology used to replicate the bounded random walk in Sutton's seminal paper. The two frameworks for using TD(λ) under both repeated and single presentation of data has been discussed. The experiment showcase some of the desirable features of TD algorithms over the conventional supervised learning when it comes to prediction problems. The experiments also reveal that at the extreme ends on TD(0) and TD(1) prediction error are worse, but a sweet spot can be found with $0 < \lambda < 1$.

REFERENCES

[1] R. S. Sutton, "Learning to predict by the methods of temporal differences" Machine Learning, vol. 3, pp. 9–44, 1988.

IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper