TD(λ)

*Replication of Sutton's Experiments [1]

1st Yuan Yang
yyang998@gatech.edu
Reinforcement Learning

***Abstract*—This work is a replication of experiments in Section 3.2 in Sutton's 1988 paper [1]. The main purpose of this part of the work is to demonstrate that even for the predictions of a simple dynamical system (*bounded random walk system*), the *temporal difference(TD)* methods could outperform the supervised-learning methods.**

I. Introduction

This work mainly addresses the learning-to-predict problems, which are using past experience to predict the future behavior of the a partially known system. For a system that is not completely understood, finding patterns in past experience could be served as the important tool for prediction. This approach is natural for solving these types of problems, as humans frequently rely on past experiences to guide their future actions for certain cases in real life.

For prediction learning, the process could be completed without supervision. Specifically, this learning is performed using methods called *temporal difference(TD)*. The main difference between *TD* and other prediction methods (e.g., *Monte Carlo*) is that the learning is directed by the difference between temporally predictions (current prediction and estimation for next state) for TD, while the learning is guided by the difference between final predicted return and actual observed result for other methods. One of the advantages of TD methods is that the computation is more efficient, since updates can occur at each time step rather than being delayed until the completion of the entire episode. Sutton's paper demonstrated that the prediction performance of TD methods in addressing arbitrary events is better than conventional prediction methods.

For the work that is replicated in this paper, the comparison between TD and supervised learning is discussed intensively for dynamical systems. Unlike TD methods, standard supervised learning ignores the sequential structure of data generated by a dynamical system, in which the states evolves and changes as time increasing. Standard supervised learning requires some ground truth labeled data to guide the learning. In order to relate TD with supervised learning in addressing Prediction Learning problems, Sutton demonstrated that prediction learning could be cast into using supervised learning with some adjustments. Since there is no labeled data to reference for in prediction learning, the data based on which a prediction must be made is served as the first item in supervised learning, while the actual outcome of this prediction is served as the second item. In another word, the final outcome acts as the outcome for each pair. The computational details on how TD can be related to classical supervised learning is introduced in next Section.

II. Computation Details and Procedures

For the implementation and computation details for comparing TD with supervised learning method, Sutton introduced a specifically designed TD paradigm to relate it to classical supervised learning paradigm (Widrow-Hoff rule) in next section.

For an observation-outcome sequence $x_1, x_2, x_3, ..., x_m, z$, x values are the observation states at certain time point and x is the reward and outcome from this specific sequence. For this sequence, each observation x_t has a corresponded prediction P_t that is generated during learning. This prediction is obtained using both observation x_t and a vector of weights that are continually being updated throughout the procedure. For a case when P_t is a linear function, it can be represented by

$$P_t = \omega^{\mathrm{T}} x_t = \sum_i \omega(i) x_t(i) \tag{1}$$

where i is the index of element in ω and x vectors.

In Sutton's paper, the learning process was represented using the update rule of weights ω. The details are discussed separately for supervised learning and TD in next two subsections *A* and *B*.

A. Supervised-learning Approach

The updating rule here for supervised learning is not computed incrementally during a sequence comparing to TD, that is ω is only updated once for each sequence and is not changed within:

$$\omega \leftarrow \omega + \sum_{t=1}^{m} \Delta\omega_t \tag{2}$$

where the accumulated weight $\Delta\omega_t$ within a sequence is only added into ω after the whole sequence has been processed.

As demonstrated in Introduction part, the final reward of a sequence acts as the outcome for each observation-outcome pair. Therefore, z is the outcome for every observation x_t in a sequence. For an observation x_t, $\Delta\omega_t$ can be calculated based on the corresponded reward z and prediction P_t :

$$\Delta\omega_t = \alpha(z - P_t)\nabla_\omega P_t \tag{3}$$

where α is the learning rate, $\nabla_\omega P_t$ is the gradient formed by the partial derivatives of P_t with respect to ω.

One thing to emphasized here is that when gradient $\nabla_\omega P_t$ equals to x_t, equation (3) and be combined with equation (1) to be simplified to Widrow-Hoff rule, which is $\Delta\omega_t = \alpha(z - \omega^{\mathrm{T}} x_t)x_t$.

B. TD(1) and TD(λ) Approaches

Unlike supervised learning, the updating rule here for TD is computed incrementally, that is ω is updated continuously during each sequence. The difference between TD and supervised learning represented using equation (3) is that $x - P_t$ component should be computed as sum of changes in predictions in TD. Therefore, the updating rule in TD can be represented by

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

where sum of changes in predictions is computed as $\sum_{t=1}^{m}(P_{t+1} - P_t)$.

Equation (4) is for TD(1) procedure. To extend this to TD(λ) procedures, an exponential weighting parameter is added to weight predictions differently, resulting in greater alterations for recent predictions. Then $\Delta\omega_t$ for TD(λ) can be calculated using:

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

where the newly added weight factor is λ^{t-k} and $0 \le \lambda \le 1$.

When λ is smaller than 1, based on equation (5), the update is more associated with recent predictions.

III. Experiments Preparation

A. Dynamical system used in experiments: random walk

A simple dynamical system called *bounded random walk* is used to perform the prediction learning experiments. The specific example of this system we used to generating result is shown in Fig. 1. In this system, a walk starts at state *D* and moves to adjacent states on the right or left side with equal chance. The walk stops when reaches terminating state *A* or *G*. In this work, Sutton predicted the probability of sequences terminating at the right side, making the reward z to be 1 for a sequence stopping at *G* and reward to be 0 for a sequence stopping at *A*. x for the 5 non-terminating states in this system are vectors x_B, x_C, x_D, x_E, x_F, in which

$$\begin{aligned} x_D &= (0,0,1,0,0)^T \\ x_C &= (0,1,0,0,0)^T \\ x_B &= (1,0,0,0,0)^T \\ x_E &= (0,0,0,1,0)^T \\ x_F &= (0,0,0,0,1)^T \end{aligned} \tag{6}$$

For example, at time t, $x_t = x_E$ if current state is at *E*.

B. Training Data Generation

100 training sets are generated for experiments to ensure reliable training results, and each set consists of 10 randomly generated sequences (every sequence starts *D* and ends at *A* or *G*). For each sequence, in addition to the states at each time point, the reward and x_t vectors are also generated. The reward for a sequence is determined by the terminating state, while the x_t vectors are computed based on the order of non-terminating states. This process is explained in details in the Jupyter notebook in my Git Repo.

One key point here is that when generating sequences, the probabilities for one state to reach its left and right states are equal (0.5 for each side), and the randomness needs to be guaranteed during this process.

C. Ideal Predictions

To measure the performance of a learning procedure, the *ideal prediction* needs to be performed to generate the true probabilities that can be used to compared to the actual learning result from the training sets. The root mean squared (RMS) error between the true probabilities and the learning res is calculated after the learning.

The experiments measure the probability terminating at *G*. For rightside termination, Sutton introduced the procedure of using *ideal prediction* to get true probability (weight) for each non-terminating state.

Based on Section 4.1 in Sutton's paper, with the Markrov theory, the true expectation of reward z when the sequence starting at state i can be represented by

$$E\{z|i\} = [(I - Q)^{-1}h]_i \tag{7}$$

where Q_{ij} is the transition probability from non-terminating state i to non-terminating state j, h_{ig} is a vector of length 5 with probabilities between 5 non-terminating states i and the rightside terminating state g, and I is identity matrix with value 1. Please refer to the Jupyter notebook for the details on how the values are computed.

IV. Experiments, Results and Discussion

A. Replicate Figure 3: Repeated Presentations Training

The TD(λ) approach in previous Section 2.B is used in this experiment, and the goal for this part of the work is to discuss the impact of λ value on learning outcome. The modification is that the total weight vector ω was not updated after each sequence, it was only updated after a training set of 10 sequences are processed. Therefore, the $\Delta\omega$ was accumulated during the processing of a whole training set, and it was added into ω afterwards.

Each training set was not processed only once, instead, it was processed repeatedly during each learning procedure until the changing pattern/trend of ω is converged. A threshold is used to determine when it is the time to stop, and the learning process for current set stops when the changes in weight vector is less than the threshold. In this experiment, I used 0.001 as the threshold.

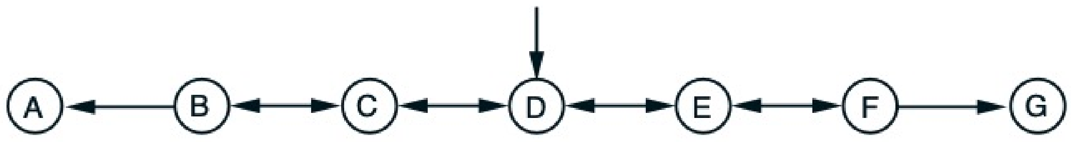
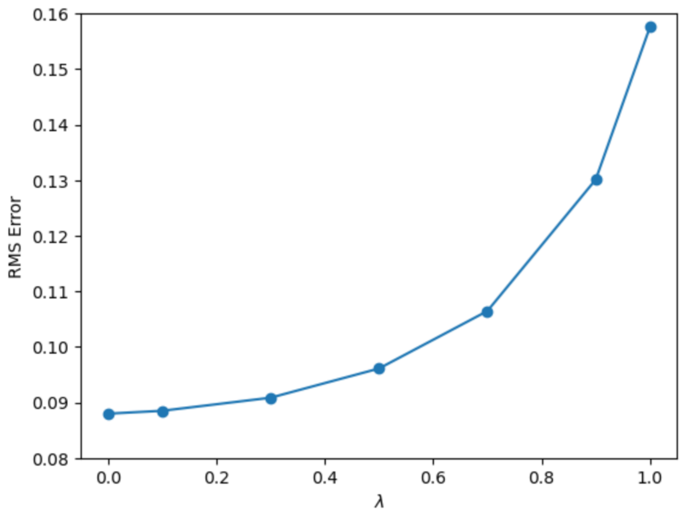
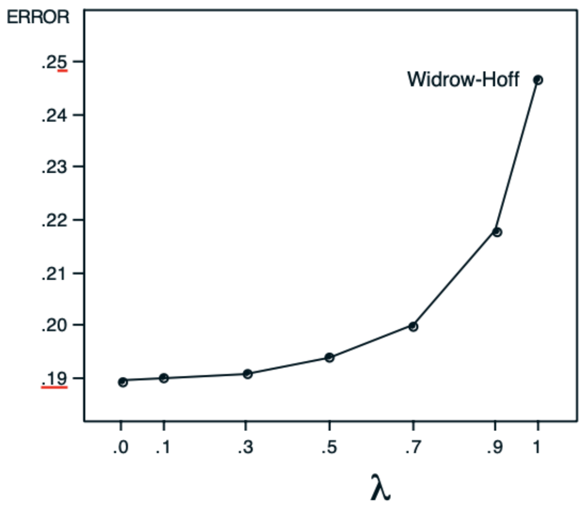


Fig. 1: Random Walk System Used: *D* is the start point for any sequence; arrows indicate that if the system has a chance to randomly walk from a state to another; *A* and *G* are the leftside and rightside boundaries for a walk to terminate at.



(a) Replicated Figure 3 in this work



(b) Original Figure 3 in Sutton's Paper

Fig. 2: Replication of Figure 3 in Sutton's Paper

For the initialization of the weight vector ω, I initialized the weight all to be 0.5 since Sutton mentioned that the final values of weights are independent from initial values for small learning rate α. Then the α I used here is 0.001.

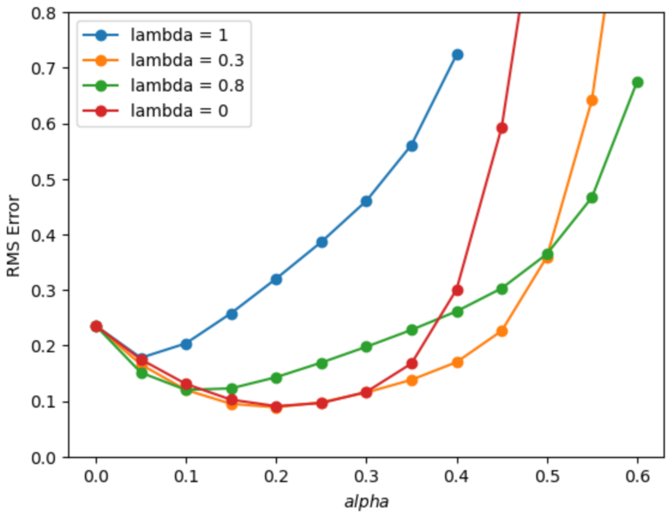
This experiment was designed to understand how the different λ values affect the RMS error between predication of the training set and the ideal prediction. The λ values I used are *0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0*. One thing to note here is that the final RMS error for each experiment with a particular λ is the averaged error across all the 100 training sets.

After performing this experiment and getting RSM error for each λ value, the generated results are plotted in Fig. 2a. For comparison, the original plot from Sutton's paper is also plotted in Fig. 2b. The replicated figure shows the similar trend as the original one as λ increasing from 0 to 1. The experiment with $\lambda = 0$ is a TD(0) procedure with best performance, while TD(1)'s performance is the worst. Recalling what was introduced previously in the Introduction and Computation sections, the experiment with $\lambda = 1$ follows the supervised learning procedure (specifically the Widrow-Hoff rule: $\Delta\omega_t = \alpha(z - \omega^{\mathrm{T}} x_t)x_t$). The Widrow-Hoff process, which represents supervised learning, resulted in the worst performance with the highest errors. Sutton's paper explains that the Widrow-Hoff method only minimizes the error on training datasets, resulting in worse generalization performance and less optimal estimation for matching future experiences.

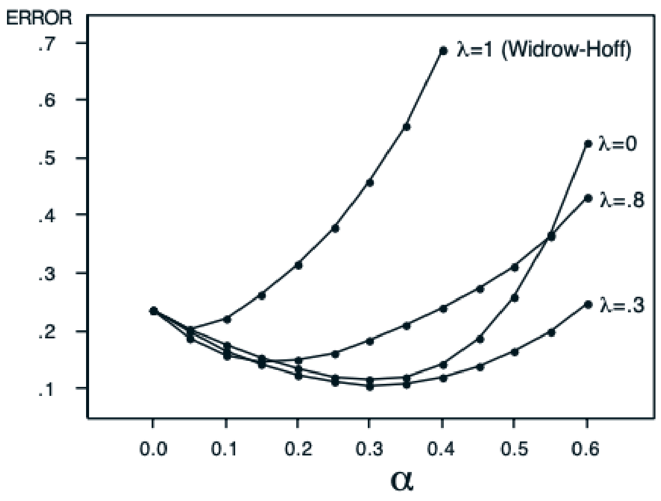
The error growths from $\lambda = 0$ to $\lambda = 1$ in both plots is at similar levels (Δ_{error} both around 0.06). The noticeable difference is that, for all λ, the errors' values in replicated plot are shifted down about 0.1 comparing to the original one. Given that the extent of error growth in both plots is approximately equal, this indicates that the impact of various λ values is at a comparable magnitude. The detected differences in absolute error values may arise from various factors, such as randomly generated training sets, initialization of weights, the value of learning rate α, and the threshold used to determine when to stop learning. In this work, it was confirmed that different training sets or initial weight values could indeed lead to variance in the error value.

B. *Replicate Figure 4: Single Presentation Training*

The objective of this section is to assess how the learning rate α impacts performance. This experiment is a simplified version of the one outlined in section 4A. In this setup, each training set was processed only once, and no need for repeated presentations until convergence. Additionally, instead of only



(a) Replicated Figure 4 in this work



(b) Original Figure 4 in Sutton's Paper

Fig. 3: Replication of Figure 4 in Sutton's Paper

updating weights after processing an entire training set, the strategy was modified to update weights after each individual sequence.

Experiments in this section consisted of using various combinations of α and λ values. Specifically, for each of λ values, different α values were paired with the current λ to serve as the input parameters: λ values are *0, 0.3, 8, 1.0*, and α values are *0.0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6*. This experiment requires no threshold since the convergence is not required, and the weight initialization remains the same as in section 4A.

After conducting this experiment and obtaining the RSM error for each combination of λ and α, the results were plotted in Fig.3a. For comparison, the original plot from Sutton's paper is included in Fig.3b. These replicated plots exhibit similar convex trends to the original ones. The TD(1) procedure generated the worst estimates at smaller α values (0 to 0.45) for all λ. This observation further confirms that all the TD(λ) methods with λ less than 0 outperforms the TD(1) method, which is the supervised learning method Widrow-Hoff.

The x-axis represents various α values. The α value at the lowest point in the convex pattern represents the optimal value of α that produces the best performance for a given λ. It was observed that the optimal α values are around 0.2 for smaller λ (0 and 0.3) in the replicated figure, while the optimal α values are around 0.3 for same λ values in original figure. The difference between my replicated results and the original ones could be due to the variation in randomly generated composition of the training sets.

C. Replicate Figure 5: Best Error Level vs. Various λ

This experiment serves as a continuation of the one in section 2B. The optimal value of α for each λ was determined by identifying the points with the lowest error on the plots

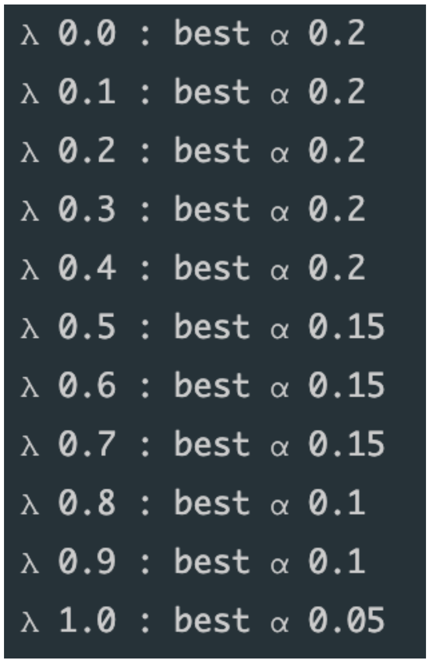
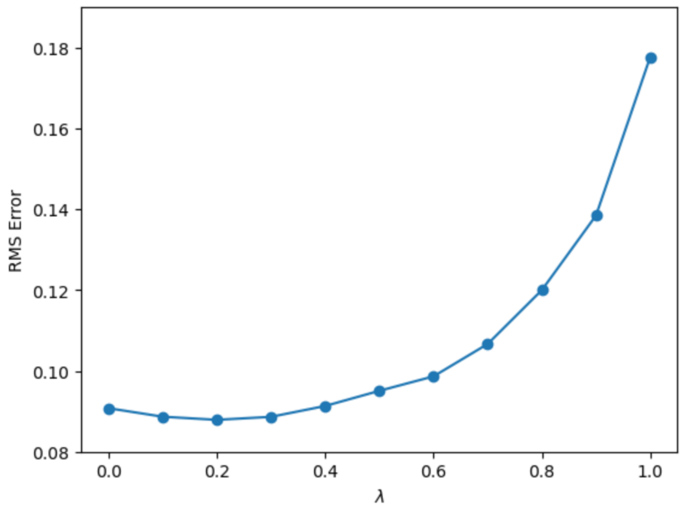

```
λ 0.0 : best α 0.2
λ 0.1 : best α 0.2
λ 0.2 : best α 0.2
λ 0.3 : best α 0.2
λ 0.4 : best α 0.2
λ 0.5 : best α 0.15
λ 0.6 : best α 0.15
λ 0.7 : best α 0.15
λ 0.8 : best α 0.1
λ 0.9 : best α 0.1
λ 1.0 : best α 0.05
```

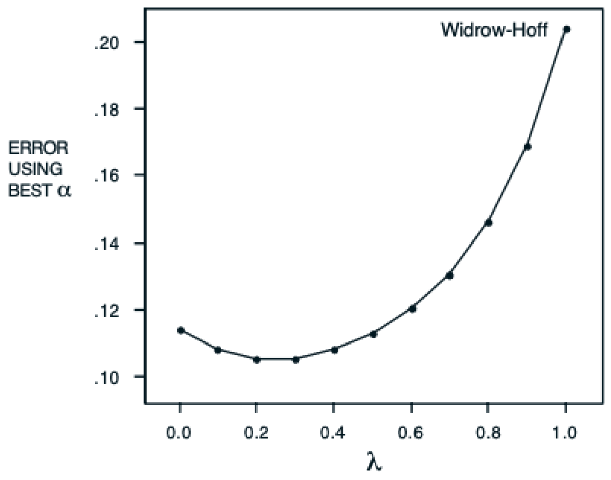
Fig. 4: λ vs. corresponded optimal α

generated in the previous experiments. For λ = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0], the obtained corresponding optimal α are listed in Fig.4. Please refer to the Jupyter notebook for details.

After collecting the combinations of λ and α that produce best estimations, the single presentation experiments are performed. The results (error vs. λ) were plotted in Fig.5a. For comparison, the original plot from Sutton's paper is included in Fig.5b. The trends are similar and absolute values of errors are also similar comparing my work with the original one. The best λ value in the replicated plot is around 0.2, while the best



(a) Replicated Figure 5 in this work



(b) Original Figure 5 in Sutton's Paper

Fig. 5: Replication of Figure 5 in Sutton's Paper

λ value in the replicated plot is also around 0.2. Still, TD(1) resulted in worst prediction performance.

Different from Fig. 2a, which was generated from experiments with repeated presentations, $\lambda = 0$ is not found to be optimal here. This is due to the fact that each training set only presented once in this experiment, and TD(0) is slower in propagating the estimations back along the sequence (slower learning).

V. Conclusions

This work replicated the prediction learning results in Sutton's paper [1] for a dynamical system called the *bounded random walk*. The replicated results exhibit similar trends compared to the original ones. Some variations in the absolute error values could be attributed by multiple factors, such as randomly generated training sets, initialization of weights, the value of the learning rate α, and the threshold used to determine when to stop learning. In general, this work successfully replicated the experiments in Sutton's paper and drew the same conclusions as the original paper. The impact of the learning rate and the λ value in TD(λ) on prediction learning was discussed, and it was suggested that TD(λ) methods outperform supervised-learning methods in providing more accurate predictions when solving prediction problems.

References

[1] Richard Sutton. "Learning to Predict by the Method of Temporal Differences," Machine Learning 3, pp. 9–44, Aug 1988.