TD(λ , μ) :Temporal Difference Learning using Future Observation

Tsuyoshi SUZUKI Nobuo INUI Yoshiyuki KOTANI
Tokyo University of Agriculture and Technology
{go, nobu, kotani}@fairy.ei.tuat.ac.jp

ABSTRACT

The prediction-learning problem is to predict future behavior by past experience. Temporal Difference (TD) learning is the learning algorithm used for a multi-step prediction-learning problem to begin in Samuel Checker playing. Sutton proposed the TD (ambda) that past observation was taken into consideration, and showed that the efficiency is better than the method that only an observation result and a final result are used for. But it is not necessary to restrict using information in past experience.

We proposed a new mechanism of learning called TD(λ , μ) that considers both observation of the past and the future[13]. This method takes relations not only with the past but also with the future into consideration as if TD(λ) is done from the future direction. There was little difference in result though the comparative experiment of the LMS, the TD (λ), the TD (λ , μ) was done by using the "world of 3×4". We stated that there was little difference in result though the comparative experiment of the LMS, the TD (λ), the TD (λ , μ) was done by using the "world of 3×4"[13]. This depends on being the environment that the "world of 4×3" is very simple. In this paper, we examined the learning experiment of this learning algorithm in Shogi, which is the complicated environment. To test the effectiveness of the learnt values, a number of matches were played between identical search engines using learnt piece values. As the result, TD(λ , μ) was the best in comparison with others.

TD(λ , μ) :未来の観測状態を考慮した TD 法

鈴木 豪　乾 伸雄　小谷善行
東京農工大学
{go, nobu, kotani}@fairy.ei.tuat.ac.jp

概　要

過去の経験から将来の振舞いの予言を学習する問題を、予言学習問題という。例えば、将棋のある局面が経験から勝ちに結びつくかどうかの予言を学習するなどである。TD(Temporal Differential:時間的差分)法は Samuel のチェッカープログラムに始まるマルチステップ予言学習問題に対する学習アルゴリズムである。Sutton は過去の観測状態を考慮する TD(λ)法を提案し、それが従来行われていた観測状態と最終結果のみからの学習法よりも効率がよいことを示した。

我々は過去と未来の両方の観測状態を考慮した TD(λ , μ)法を提案した[13]。TD(λ , μ)法は過去と未来の両方の観測状態を考慮して学習を行うため、TD(λ)法よりも効率が良いことが期待される。先の発表では "3× 4 の世界" を使って、LMS 法、TD(λ)法、TD(λ , μ)法の比較実験を行ったが、大きな差は見られなかった。これは "4× 3 の世界" が非常に単純な環境であることによる。本稿では、将棋というより複雑な環境下でこれらの学習アルゴリズムを使って学習実験を行った。学習で得られた駒価値を使った対戦実験の結果、TD(λ , μ)法は、LMS 法、TD(λ)法の中で最も高い勝率を上げた。

1 Introduction

The problem that the prediction of the future behavior is learned from the past experience is said as the prediction-learning problem. For example, it is to learn a prediction whether the position with Shogi is connected with the victory from the experience. In multi-step prediction problems, correctness is not revealed until more than one step after the prediction is made. It is a multi-step prediction problem to expect the victory or defeat of the Shogi, because it doesn't know the correctness of the prediction until the game completion.

Simple algorithm for prediction learning is the LMS (Least Mean Square) method, which was proposed by Widrow and Hoff (1960). This algorithm is learned only from the observation and the final result. Temporal Difference (TD) learning is the learning algorithm used for a multi-step prediction-learning problem to begin in Samuel Checker playing. Sutton (1988) proposed the TD(λ) methods that past observation was taken into consideration, and showed that the efficiency was better than the LMS. TD(λ) was applied to learning of the game, and Tesauro's TD-Gammon reached the strength of the master level.

Conventional TD(λ) learns a prediction by using the only past information. While TD(λ , μ) method considers both observation of the past and the future. Efficiency may be good when the prediction is not only from the past and from the future is used. If an observation sequence was given, the learning efficiency may be good when the prediction is not only from the past and from the future is used. For example, it is possible that it guesses the past situation from the future situation in the case of Shogi. In other words, the past situation is restricted at present by the situation. Therefore, TD(λ , μ) can be expected to learn more efficiently than the TD (?) under the complicated environment. In TD(λ , μ), the relations with the past predictions are being taken into consideration by using the gradient vector that weighted the exponential form same as TD(λ), and the relations with the future predictions are being taken into consideration by using the gradient vector that weighted the exponential form. TD(λ , μ) is the generalization of the TD (?), because if μ =0 then TD(λ , μ) and TD (?) are the same rule.

In Chapter 2, we describe LMS, TD(λ), and TD (?, μ). We describe how to learn the pieces values in chapter 3. And we describe those results in chapter 4.

2 Learning algorithms

In this chapter, we describe about LMS and TD(λ) at first. And we describe TD (?, μ) that considers the observation of the future.

2.1 LMS and TD(λ)

We suppose that observation data at the time 1,2,3···,n are $x_1, x_2, \cdots, x_n$, and a final result is z . And, we suppose that using the observation data and the parameter w makes the prediction $P(x,w)$. Learning adjusts the parameter w of the prediction.

Then LMS method is learned only from the observation and the final result. A parameter is updated with the following formula:

$$w \leftarrow w + \sum_{t=1}^{n} \Delta w_t$$

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

Here, a positive constant α is called a learning rate. LMS is the procedure of updating a parameter to make a prediction and the least mean error of the actual result the smallest:

$$Err = \frac{1}{n} \sum_{t=1}^{n} \left(z - P(x_t, w)\right)^2$$

As for a fault of the LMS, it is ignored that each data that it is observed is not independence. For example in the case of Shogi, there are intimate relations of a position with the position until it reaches it. Because these relations are ignored, LMS learns slowly.

TD(λ) is based on the supposition that present observation condition influences past observation condition and happens, and past condition is being taken into consideration at the time of learning. TD(λ) update the parameter from the difference between the predictions. And, relation with the past is being taken into consideration by using the gradient vector that weighted the exponential form. A parameter is updated with the following formula:

$$w \leftarrow w + \sum_{t=1}^{n} \Delta w_t$$

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

Here, λ is the positive constant that relations between the predictions are expressed. Especially, it knows that it corresponds to the LMS with TD(1) made $\lambda = 1$.

2.2 TD(λ , μ)

TD(λ) is based on the supposition that present observation condition influences past observation condition and happens, and past condition is being taken into consideration at the time of learning. As for the future observation condition, there is relation in present observation condition in the same way as present observation condition has relations with the past observation condition. And we can think that it can guess condition at present from the future. TD(λ , μ) is based on the idea that better learning is done by using both relations of the past and the future.

When relations with the future are taken into consideration, it thinks about the method of the various considerations. We will take it into consideration in the same way as the TD(λ) by using the gradient vector which weighted the exponential form. A parameter is updated with the next formula:

$$w \leftarrow w + \sum_{t=1}^{n} \Delta w_t$$

$$\Delta w_t = \alpha(P_{t+1} - P_t)\left(\sum_{k=1}^{t} \lambda^{t-k} \nabla_w P_k + \sum_{k=t+1}^{n} \mu^{k-t} \nabla_w P_k\right)$$

TD(λ , μ) is the generalization of the TD(λ), because if μ =0 then TD(λ), and TD(λ , μ) are the same algorithms. Also TD(0,0) are same as LMS.

TD(λ , μ) takes both condition of the past and the future into consideration by using the gradient vector which weighted the exponential form. That can be computed incrementally as in TD(λ). To do this, we calculate from the past to the future, and from the future to the past separately. The caluculation from the past to the future are following:

$$e_t \equiv \sum_{k=1}^{t} \lambda^{t-k} \nabla_w P_k$$

$$e_{t+1} = \sum_{k=1}^{t+1} \lambda^{t+1-k} \nabla_w P_k = \nabla_w P_{k+1} + \sum_{k=1}^{t} \lambda^{t+1-k} \nabla_w P_k = \nabla_w P_{k+1} + \lambda e_t$$

And the caluculation from the future to the past are following:

$$e'_t \equiv \sum_{k=t+1}^{tm} \mu^{k-t} \nabla_w P_k$$

$$e'_{t-1} = \sum_{k=t}^{tm} \mu^{k-t+1} \nabla_w P_k = \nabla_w P_{k+1} + \sum_{k=t+1}^{m} \lambda^{k-t} \nabla_w P_k = \nabla_w P_{k+1} + \lambda e'_t$$

We can update incrementally using e_t and e'_t.

3 The Experiments

We examined the experiment of learning of piece values in Shogi by using LMS, TD (?) and TD (?, µ). Game records were used for learning. Game records were made by the self-play of Kakinoki Shogi Ⅲ (level 1:it is the weakest). A weak level was used in the experiment to try to reduce influences by other factors because only piece value was being used for the evaluation function.

The setup of a problem to learn is made the following. The setup of a problem to learn is made the following. The difference vector of pieces is defined as follows:

$$x_i = (\textit{the}\text{ number of the piece } i \text{ of the black}) - (\textit{the}\text{ number of the piece } i \text{ of the white})$$

$$i \in \{\text{歩},\text{香},\text{桂},\text{銀},\text{金},\text{角},\text{飛},\text{と},\text{成香},\text{成桂},\text{成銀},\text{馬},\text{竜}\}$$

$$x = (x_{\text{歩}}, x_{\text{香}}, \cdots, x_{\text{竜}})$$

States of positions were observed by using this difference vector:

$$x^1, x^2, \cdots, x^{m-1}, x^m = z\,(\text{final result})$$

Here, m is the number of the states that is observed during a game. The weight vector that value of each piece was arranged is defined as follows:

$$w = (w_{\text{歩}}, w_{\text{香}}, \cdots, w_{\text{竜}})$$

We define the prediction with the following formula:

$$y = x \cdot w$$

$$P(x,w) = \frac{1}{1+\exp(-y)}\,.$$

The differences of the piece values (the difference in evaluation value of the position) are mapped to interval [0,1]. Using

$$\frac{\partial P}{\partial w_i} = \frac{\partial P}{\partial y}\frac{\partial y}{\partial w_i} = P(1-P)w_i\,,$$

weight vectors are updated with the following formula:

$$\Delta w = \alpha(z - P^t)P^t(1-P^t)x_t \quad (\textit{for LMS})$$

$$\Delta w = \alpha(P^{t+1} - P^t)\sum_{k=1}^{t}\lambda^{t-k}P^t(1-P^t)x_k \qquad (\textit{for TD}(\lambda))$$

$$\Delta w = \alpha(P^{t+1} - P^t)\times\left(\sum_{k=1}^{t}\lambda^{t-k}P^k(1-P^k)x_k + \sum_{k=t+1}^{n}\mu^{k-t}P^k(1-P^k)x_k\right) \qquad (\textit{for TD}(\lambda,\mu))$$

Table 3.1 shows the parameters used in the learning.

Table 3.1 Parameters

	α	λ	μ
LMS	0.01	-	-
TD(λ)	1.00	0.80	-
TD(λ , μ)	1.00	0.80	0.20

It was converge in the unrealistic value when a of LMS was set up in 1.0 in the same way as TD. For that reason, a is set up in the little value in the case of LMS. The value of ? and µ was decided as a result of doing the spare experiments. α is made to decrease together with the number of updating in each algorithm :

$$\alpha_n = \alpha\exp\left(-\frac{n}{1000}\right)$$

Here, n is number of updating. Learning stopped when the updating of the weight was 3000 times.

4 Results

In this chapter, we present results of the learnt piece values. And to test the effectiveness of the learnt values, a number of matches were played between identical search engines using learnt piece values.

4.1 Weight traces and learnt values

Figure4.1 shows the weight for pieces learned by LMS. It is a gentle learning curve of the LMS compared with (?), and TD (?, μ), because the a was set up small. We can see that the relative ordering of the pieces has been decided at the early grade of learning. Ratios between values of pieces are being adjusted after that.

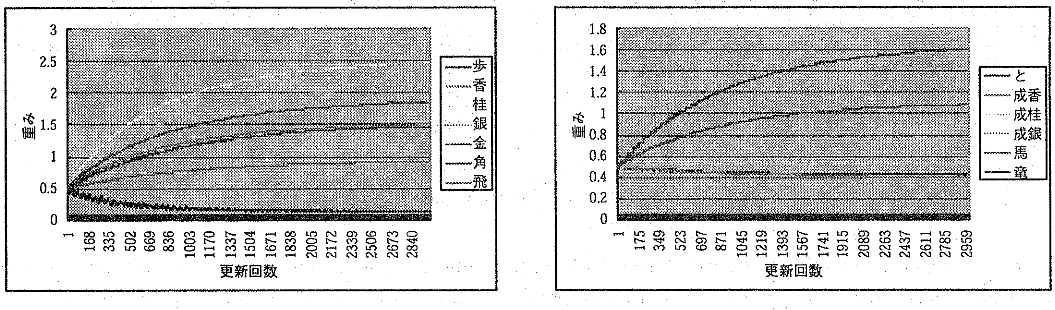


Figure 4.1 Weight traces (LMS)

Figure4.2 shows the weight for pieces learned by TD(λ). We can see that the relative ordering of the pieces has been decided after about 800 games. We guess that it can't learn about the promoted pieces, because it seldom appears.

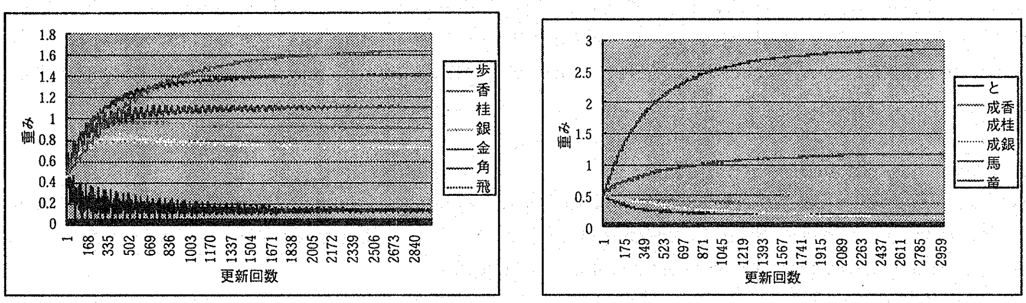


Figure 4.2 Weight traces (TD(λ))

Figure4.3 shows the weight for pieces learned by TD(λ). The learning curve of TD(λ , μ) and TD(λ) resemble.

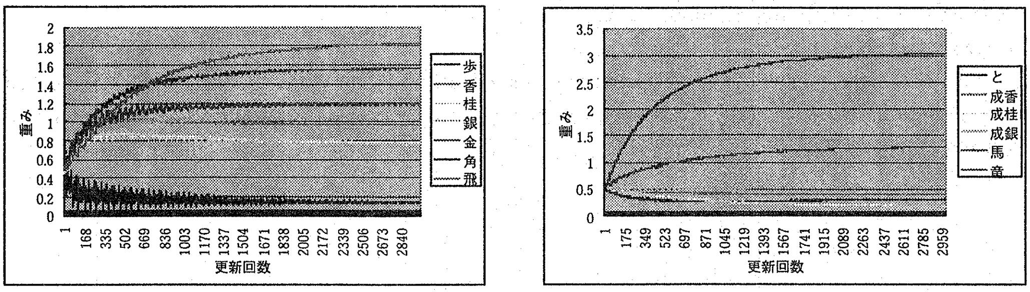
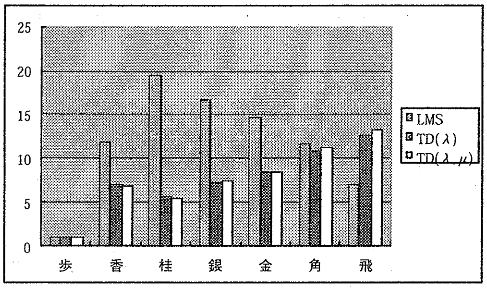


Figure 4.3 Weight traces (TD(λ , μ))

Figure 4.4 Shows relative values for the pieces (We defined 歩=1). The learning result of LMS is greatly different from others. On the other hand, TD (?) and TD (?, μ) are the results that it often looks alike. We guess that it can't learn about the promoted pieces, because used data were only 50 and promoted pieces seldom appear.



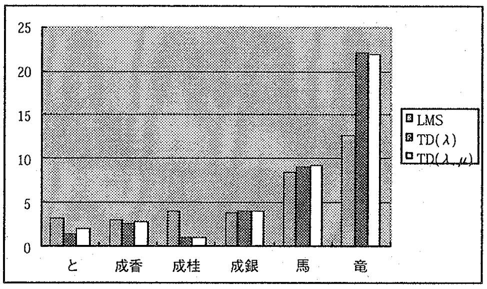


Figure 4.4 Learnt values

The value learned with LMS is greatly different from the value that a human being thinks about. The value of the piece learned with TD (?) and TD (?, μ) isn't greatly different from the value that a human being thinks about. But, the value of the 香 is a little big. The value of the dragon shows especially big value. We guess that it is because the data of the lowest class person are being used. When a beginner plays Shogi, the power of the 竜 has an influence very greatly.

It was not realistic value as a result of doing learning as a=1.0 in the LMS. When a was made small, it was the same result. For that reason, it was set up in a=0.01 and the extremely little value in this experiment. But, it couldn't still learn well. There was little difference in result though the comparative experiment of the LMS, the TD (?), the TD (?, μ) was done by using the "world of 3×4" [?]. We aren't thought that LMS works well under a complicated environment like Shogi. It depends on having set up μ in the little value that the result of TD (?) and TD (?, μ) looks alike.

4.2 Matches to Test the Learnt Values

To test the effectiveness of the learnt values in Shogi, a number of matches were played between identical search engines using learnt values. Shogi program "Sexy-AI-chan" being made in Kotani laboratory was used for the experiment. An experiment went on the following condition:

·use alpha-beta pruning
·use iterative deepening search
·use search depth is three-play
·use capture extension
·use check extension
·factors of evaluation function are:
  piece value, distance form 王, effect near 王
·piece values are learnt values

Table 4.1 shows the number of matches between learnt values. As for the matches of TD and LMS, TD beat it greatly. Therefore, it is stopped by 200 matches. Because a match result was close, it was made to play about 1000 games with the TD (?) and the TD (?, μ).

Table 4.1 Number of matches

		LMS	TD(λ)	TD(λ , μ)
LMS	black		219	214
	white		235	211
TD(λ)	black	235		505
	white	219		545
TD(λ , μ)	black	211	545	
	white	214	505	

Table 4.2 shows the match results. A result is shown in the percentage of victories. Each upper raw is the percentage of victories of the black move turn. Each middle raw is the percentage of victories of the white move turn. Each bottom raw is the percentage of victories of the total.

Table 4.2 Match results

		LMS	TD(λ)	TD(λ , μ)
LMS	black		0.062	0.014
	white		0.013	0.033
	total		0.024	0.024
TD(λ)	black	0.938		0.448
	white	0.987		0.445
	total	0.976		0.446
TD(λ , μ)	black	0.986	0.552	
	white	0.967	0.555	
	total	0.976	0.554	

There was a big difference in the match with the LMS and others. On the other hand, TD (?, μ) beat it 55% in the game of TD (?) and TD (?, μ). Though a game experiment was done several times, a result was the same, and the percentage of victories of TD (?, μ) was the highest.

5 Conclusions and Future Work

We examined the experiment of learning of piece values in Shogi by using LMS, TD (?) and TD (?, μ). The percentage of victories of the piece value that learnt with TD (?, μ) was the highest. We showed the possibility that TD (?, μ) was more effective than TD (?). There is the following thing as a future subject.

- There are few game records used for learning with 50 games. It is necessary to increase this number further.
- We did the learning experiment of only piece value of Shogi. The evaluation function of Shogi is more complicated.
- It tries to apply TD (?, μ) to other games as well like Shogi.
- We don't show that TD (?, μ) converges to the ideal prophecy.

We think that TD (?, μ) is more effective than conventional TD (?) under the complex environment.

References

[1] Beal, D.F. and Smith, M.C. (1997). Learning Piece Value Using Temporal Differences. ICCA Journal, Vol. 19, No. 3. Pp.147-151.

[2] Baxter, J. Tridgell, A. and Weaver, (1998). Experiments in Parameter Learning Using Temporal Differences. ICCA Journal, Vol. 20, No. 3. Pp.147-161.

[3] Sutton, R.S. (1988). Learning to Predict by the Methods of Temporal Differences. Machine Learning, Vol. 3, pp.9-44.

[4] Sutton, R.S. and Barto, A.G. (1998) Reinforcement Learning:An Introduction. MIT Press.

[5] Tesauro, G. (1995). Temporal Difference Learning and TD-Gammon. Communications of the ACM, Vol. 38, No. 3.

[6] Jaakkola, T. and Jordan, M.I. (194). On the Convergence of Stochastic Iterative Dynamic Programming Algorithms.
[7] Dayan, P. (1992). The convergence of TD(λ) for general λ . Machine Learning 8, 341-362.
[8] Peng J., and Williams R.J. (1993). TD(λ) convergence with probability 1. Department of Conputer Science preprint, Northeasten University.
[10] Sutton, R.S. (1994). On Step-Size and Bias in Temporal-Difference Learning. Preceedings of the Eighth Yale Workshop on Adaptive and Learning Systems. 91-96.
[11] Schraudolph, N. , Dayan, P. and Land, M. (1996) Temporal Difference Learning of Position Evaluation in the Game of Go. In Jack Cowan Gerry Tesauro, and Josh Alspector, editors, editors, Andvances in Neural
[12] Information Processing systems 6, San Fransisco, 1994. Morgan Kaufman.
Tesauro, G. (1992). Practical Issues in Temporal Difference Learning. Machine Learning 8, 257-278.
[13] Suzuki,T. , Kotani,Y. (2000). About TD Learning using Future Observation. 2000-GI-3. 17-24 (In Japanese).