Empirical Study of TD$_\gamma$ Reinforcement Learning algorithm

Kirill Bobyrev
Optimization Class Project. MIPT

Reinforcement Learning at a glance

Reinforcement Learning is an area of machine learning concerned with how *agents* should take *actions* in an *environment* so as to maximize the *cumulative reward*.

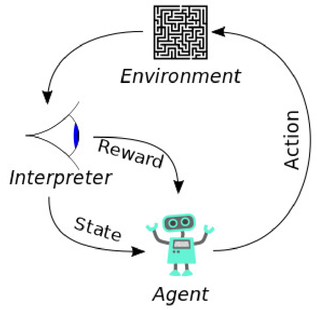
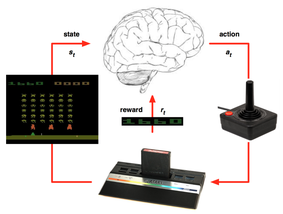


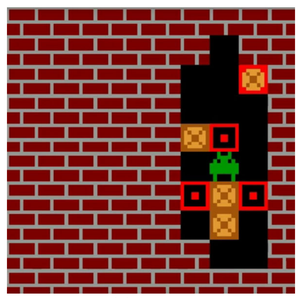
Figure 1: Reinforcement Learning world model

Reinforcement Learning problem is typically modeled by Markov Decision Process:

- $\mathcal{S}$ — set of states, it can be finite or infinite, discrete or continuous
- $\mathcal{A}$ — set of actions available to the agent
- $P_a(s, s') = P(S_{t+1} = s' | S_t = s, A_t = a)$ — probabilities of action a in state s leading to state s'
- $r_a(s, s') = r(S_{t+1} = s' | S_t = s, A_t = a)$ — numerical reward distribution, a rule incorporated in the environment which indicates which reward is given to the agent upon taking action a in state s and resulting in state s'
- $\gamma \in [0, 1]$ — discount factor which controls the relative difference in importance between the future and present rewards



(a) Atari 2600 games



(b) Sokoban

Figure 2: Sample environments

It is often assumed that the transitional probabilities and reward distribution are not directly available to the agent. The goal of the Reinforcement Learning agent is to maximize $\sum_{t=0}^{\infty} \gamma^t R_{a_t}(S_t, S_{t+1})$ by finding the optimal policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$. In order to achieve this goal, the agent typically learns value function $v_\pi(s) = \mathbb{E}[R] = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R_t | S_0 = s]$

TD(γ) Algorithm

The main contribution of the studied paper is introducing the TD(γ) family of algorithms, which eliminates the λ parameter. It is shown that TD(γ) outperforms current state-of-the-art algorithms, such as TD(λ). The objective of any value estimation algorithm is to minimize the following expression given a set of trajectories sampled using a certain policy and a set of hyperparameters: $E(\theta) = \frac{1}{2} \sum_{\tau \in T} \sum_{t=0}^{l_\tau - 1} \left(R^\gamma_{s_t^\tau} - \check{V}_\theta(s_t^\tau)\right)^2$.

Given: A discount factor γ, set of trajectories T, learning rate α

Ouput: θ — parameters of value function approximation

Let $\theta \leftarrow \mathbf{0}$. (arbitrary $\check{V}$ parameters initialization)
for each trajectory $\tau \in T$ **do**
  Store ϕ_0 in memory (ϕ_t is feature vector of state in τ at time step t)
  for $u = 1$ to l_τ **do** (l_τ stands for length of episode τ)
    Store ϕ_u and R_{u-1} in memory
    $\delta \leftarrow \mathbf{0}$
    for $t = 0$ to $u - 1$ **do**
      $\boldsymbol{a} \leftarrow \phi_t - \gamma^{u-t} \phi_u$ — accumulate trace
      $b \leftarrow \sum_{i=t}^{u-1} \gamma^{i-t} R_i$ — accumulate true rewards sequence for last $u - t - 1$ steps
      $w(u - t, l_\tau - t) \leftarrow \frac{(\sum_{i=1}^{u-t} \gamma^{2(i-1)})^{-1}}{\sum_{u=1}^{l_\tau - t} (\sum_{i=1}^{u} \gamma^{2(i-1)})^{-1}}$ — setup a custom weighting function similar to TD(λ)'s
      $\boldsymbol{\delta} \leftarrow \boldsymbol{\delta} + w(u - t, l_\tau - t)[\boldsymbol{\theta} \cdot \boldsymbol{a} - b]\phi_t$
    end for
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \boldsymbol{\delta}$ (perform semi-gradient parameter update)
  end for
  Discard all ϕ and R from memory
end for

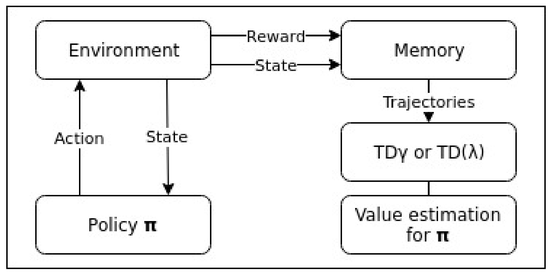


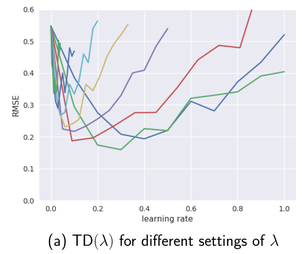
Figure 3: High-level overview of the experiment structure

Importance of value estimation

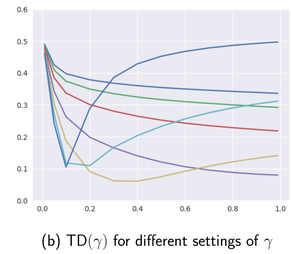
Even though the field has shifted towards *Q-Learning* which approximates $q_\pi(w, a) = \mathbb{E}[R] = \mathbb{E}[\sum_{t=0}^{\infty} R_t | S_t = s, A_t = a]$, TD($\gamma$) can be used to improve the results of state-of-the-art algorithms. By definition, $v_\pi(s) = \sum_{a \in \mathcal{A}} q_\pi(s, a)$ and hence knowing the dynamics of the environment one could efficiently navigate it using value function. However, in most cases the transitional probabilities are not known to the agent. More sophisticated methods like model-based Reinforcement Learning are able to deal with that issue. Despite the additional complexity, such methods show good performance in cases where thorough planning is crucial for the success of chosen approach. A great example of such problem would be Sokoban where researchers used the environment model to approximate model dynamics.

Experiment

Both TD(λ) and TD(γ) are benchmarked on rather simple environment 19-Random-Walk. 19-Random-Walk consists of 19 non-terminal states situated on a straight line and two terminal states at each end (one with -1 and one with $+1$ reward, the available actions are: move left and move right). Such choice is satisfied by the ability to solve Bellman equation fairly easy and obtain true values for each state given a specific policy. The policy to be evaluated is a random policy which uniformly samples action from $\mathcal{A}$. $TD(\lambda)$ and $TD(\gamma)$ produce θ so that these expectations can be approximated using $\check{V}$. Both algorithms are minimizing RMSE$(v^*_\pi, \theta) = \sqrt{\sum_s (v^*_\pi(s) - \check{V}(\theta))^2}$ and thus the lower RMSE, the better algorithm's performance.



(a) TD(λ) for different settings of λ



(b) TD(γ) for different settings of γ

Figure 4: Performance comparison for the same number of episodes (100) and different hyperparameters settings

As shown on Figure 4, TD(γ)'s performance is superior even for non-optimal hyperparameters settings . However, the effect of having lower variance of value estimate has its cost: while TD(λ) does not require the episode to terminate before operating and even allows online setting, TD(γ) does not have any of mentioned advantages which makes it unfeasible for continuous tasks.

Conclusion & Future Work

Novel TD(γ) algorithm was studied in the scope of this project. It was proven to be able to outperform its commonly used precursor. Few key improvements are yet to be made: TD$_\gamma(C)$ is another algorithm introduced by the studied paper, which can be improved by introducing automated ways to choose C and reducing the hyperparameter search space. TD(γ) can be studied in more challenging settings and environments in order to test its feasibility in state-of-the-art tasks.

References

[1] George Konidaris, Scott Niekum, Philip S. Thomas, *TD$_\gamma$: Re-evaluating Complex Backups in Temporal Difference Learning*, NIPS, 2011.

[2] Richard Sutton and Andrew Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, Second edition, 2018.

[3] Google Colab — IPython Notebook with experiments based on Shangtong Zhang's work, omtcvxyz/optimization-class-project — materials.