2021 43rd Annual International Conference of the
IEEE Engineering in Medicine & Biology Society (EMBC)
Oct 31 - Nov 4, 2021. Virtual Conference

Kernel Temporal Difference based Reinforcement Learning for Brain Machine Interfaces*

Xiang Shen, *Student Member, IEEE*, Xiang Zhang, *Student Member, IEEE*, and Yiwen Wang, *Senior Member, IEEE*

Abstract— **Brain-machine interfaces (BMIs) enable people with disabilities to control external devices with their motor intentions through a decoder. Compared with supervised learning, reinforcement learning (RL) is more promising for the disabled because it can assist them to learn without actual limb movement. Current RL decoders deal with tasks with immediate reward delivery. But for tasks where the reward is only given by the end of the trial, existing RL methods may take a long time to train and are prone to becoming trapped in the local minima. In this paper, we propose to embed temporal difference method (TD) into Quantized Attention-Gated Kernel Reinforcement Learning (QAGKRL) to solve this temporal credit assignment problem. This algorithm utilizes a kernel network to ensure the global linear structure and adopts a softmax policy to efficiently explore the state-action mapping through TD error. We simulate a center-out task where the agent needs several steps to first reach a periphery target and then return to the center to get the external reward. Our proposed algorithm is tested on simulated data and compared with two state-of-the-art models. We find that introducing the TD method to QAGKRL achieves a prediction accuracy of $96.2\% \pm 0.77\%$ (mean $\pm$ std), which is significantly better the other two methods.**

Clinical Relevance—**This paper proposes a novel kernel temporal difference RL method for the multi-step task with delayed reward delivery, which potentially enables BMI online continuous decoding.**

I. Introduction

Brain-machine interfaces (BMIs) [1] enable paralyzed people to control external devices by interpreting their motor intentions and translating them into commands with a decoder [2]. Since reinforcement learning (RL) can train the mapping from neural activity to movements without an explicit signal, instead relying on a reward from the environment, it is a more promising way for disabled people to learn to control a prosthesis through trial and error. Several RL methods have been proposed to learn the state-action mapping for BMIs [3]–[6]. However, existing algorithms can only deal with tasks with immediate reward delivery. For a complex experiment design where subjects need to take several steps to accomplish the task but are only rewarded or punished at the end of the trial, current RL algorithms cannot assign the reward over time to learn the task. Incorporating a temporal difference method into RL is one the options to solve such a temporal credit assignment problem.

Temporal difference learning, especially TD (λ), provides an efficient learning procedure for RL. Brosch *et al.* [7] applied a TD Sarsa-style [8] learning signal to AGREL to tackle delayed rewards. However, this Sarsa-style AGREL (SAGREL) is as sensitive to the initializations as AGREL. Moreover, one-step Sarsa method can barely handle the credit assignment over time when the task becomes complex and neural data involves noise, thus the algorithm may take a long time to converge [9]. Bae *et al*. introduced kernel TD(λ) to Q-learning (Q-KTD) for neural decoding in the RLBMI framework [10], [11]. It utilizes a kernel network to achieve a global linear structure but still uses an instant reward for multi-step task training. Moreover, the ε-greedy policy is applied to select actions, which may cause it to become trapped in the local minima over time without immediate reward delivery. This policy exploits current knowledge to take an action associated with the maximal reward with possibility $1-\varepsilon$ (ε is a small constant, e.g., 0.01), and all the other actions are chosen with a very small, uniform and independent possibility ε. In this way, some actions with a lower value may seldom be explored, even if they are actually better in the later training sessions. To achieve a fast and stable decoding, we adopt Quantized Attention-Gated Kernel Reinforcement Learning (QAGKRL) [5], which is more efficient to learn the neural state-action mapping with a softmax policy, to incorporate with TD (λ) for the multi-step task.

In this paper, we propose to embed a temporal difference method, namely TD (λ), into QAGKRL (TD-QAGKRL) to assign the credit over time to learn the state-action mapping in the multi-step task. We simulate a center-out task where the

* Research supported by grants from Shenzhen-Hong Kong Innovation Circle (Category D) (No. SGDX2019081623021543), the National Natural Science Foundation of China (No.61836003), Sponsorship Scheme for Targeted Strategic Partnership (FP902), special research support from Chao Hoi Shuen Foundation, Seed fund of the Big Data for Bio-Intelligence Laboratory (Z0428) from HKUST.

Xiang Shen is with the with Department of Electronic and Computer Engineering Hong Kong University of Science and Technology, Hong Kong (e-mail: xshenai@ connect.ust.hk).

Xiang Zhang, and Yiwen Wang are with the Department of Electronic and Computer Engineering, Yiwen Wang is also with the Department of Chemical and Biological Engineering, the Hong Kong University of Science and Technology, Hong Kong. Yiwen Wang serves as the corresponding author (phone: 852-2358-7053; fax: 852-2358-1485; e-mail: eewangyw@ust.hk).

subject needs to first reach the target and then go back to the center to accomplish the task to get the external reward. Here, we apply TD-QAGKRL as a decoder to translate simulated M1 signals into movements. The reward is only given at the end of the trial and the decoder needs to backpropagate the reward to the previous steps to learn the mapping. The decoding performance of TD-QAGKRL is compared with those of two existing methods (SAGREL and Q-KTD) based on the correctness of the ratio of the predicted action sequence with regard to the ground truth. The rest of the paper is organized as follows: data simulation and the experiment diagram are presented in Section IIa; the details of the TD-QAGKRL algorithm are explained in Section IIb; Section III visualizes the simulation data structure and demonstrates the training and testing results for the three algorithms; in the last section, Section IV, conclusions are presented.

II. METHODS

A. Simulation design and data generation

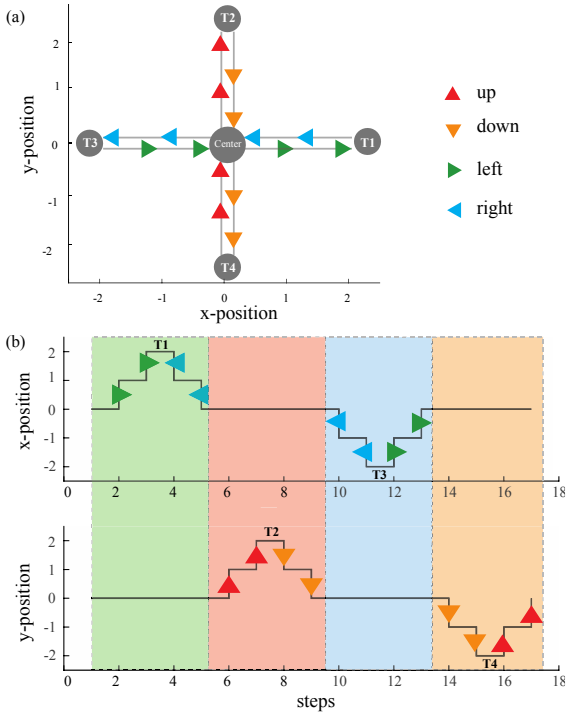

Figure 1. (a) Experiment diagram. (b) x and y positions of the trajectory in the four trials.

We simulate a center-out experiment as [12], which is commonly used in the BMI paradigm, to test the proposed temporal difference-based RL algorithms. In this task, a cursor will appear in the center of the screen at the beginning of each trial. A trial is successful if the subject first moves the cursor to one of four periphery targets and then back to the center within the predefined steps. Given the features of the trajectory, reaching actions are categorized into four classes: up down, right and left. We form the trials by arranging actions in a temporal sequence. The four actions are represented by triangles with different colors and the targets are labelled as grey circles in Fig.1 (a). Fig. 1 (b) shows the x and y positions of the 2-D trajectories in the four trials, which are shown in the sequence of [T1, T2, T3, T4], respectively.

To generate simulated spikes of the artificial neurons, we adopt the following exponential approximation function[13]:

$$f(\theta) = b + a\,exp\big(k\,cos(\theta - \mu)\big) + n\,, \qquad (1)$$

where θ is the cursor movement angle, b, a and k denote the background firing, height and shape of the tuning function respectively, μ is the preferred direction of a neuron, and n is the Gaussian noise. We define b=0, a=1.6, and k=1 in the simulation. We generate four neurons with preferred directions μ=0◦, 90◦, 180◦ and 270◦, which are the same as the four respective targets. Gaussian noise n is explored to make the simulated data mimic the neural data in real sensoria. The simulated data are generated every 100 ms.

B. Action decoding and parameter updating for TD-QAGKRL

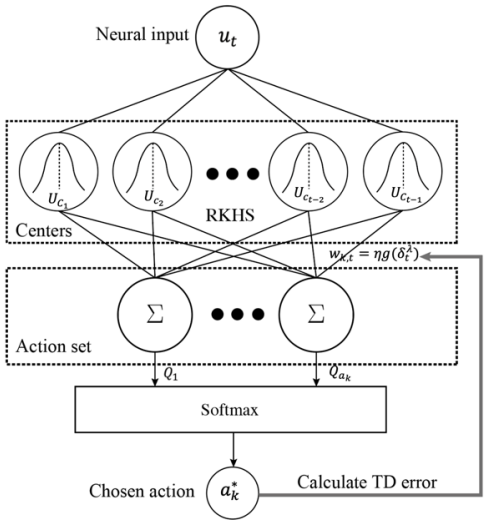

Figure 2. The structure of TD-QAGKRL.

The structure of TD-QAGKRL is shown in Fig. 2. The neural input $u_t \in R^{1\times 5}$ (four simulated neurons with a bias) is transformed to a Reproducing Kernel Hilbert Space (RKHS) by a kernel method, $\kappa(u_t, u_j) = <\phi(u_t), \phi(u_j)>$, which is commonly used as a Gaussian kernel:

$$\kappa(u_t, u_j) = \exp\left(-\frac{\|u_t - u_j\|^2}{2\sigma^2}\right), \qquad (2)$$

where σ is the kernel size, which decides the flatness of the Gaussian kernel. Then the action value is computed by linear combination with the weights as follows:

$$Q_k(u_t) = \sum_{j=1}^{t-1} w_{k,j}\langle\phi(u_t), \phi(u_j)\rangle = \sum_{j=1}^{t-1} w_{k,j}\,\kappa(u_t, u_j), \quad (3)$$

where $w_{k,j}$ is the coefficient between the jth Gaussian kernel center in the RKHS and the kth action.

An RBF network has an inherently growing structure as it allocates a new center for each coming data sample, which causes a linearly growing computational complexity. TD-QAGKRL adopts an effective quantization approach to decrease the number of kernel centers [5]. We explore an

optimal quantization threshold ξ_U according to the distribution of the Euclidean distances between the pairs of input vectors. The output action values can be calculated as follows:

$$Q_k(u_t) = \sum_{j=1}^{|C_{t-1}|} w_{k,j}\, \kappa(u_t, u_j^q) \tag{4}$$

$$d_{min}^C = \min_{1\le k\le |C_{j-1}|} \| u_j - C_k \|, \tag{5}$$

where $|C_{t-1}|$ is the final size of the centers, including input patterns of the preceding $(t-1)$ samples and u_j^q is the jth center after quantization, whose minimal distance $d_{\min}^C$ in (5) from all the previous centers is larger than ξ_U.

After the action values are calculated, we will probabilistically choose the action $a_t{}^*$ based on the softmax policy, defined as $P(Z_{a_t}=1) = \frac{\exp(Q_{a_t}(u_t)/\tau)}{\sum_{a_t'=1}^{K}\exp(Q_{a_t'}(u_t)/\tau)}$, where τ is the temperature parameter and K is the action set (four actions). If the decoded action sequence successfully reaches the goal of the task, the decoder will get a reward r =1. Otherwise, it receives no reward r = 0. For this multi-step task, the reward is only given at the end of the trial, and no intermediate step will receive any external reward.

The TD-QAGKRL is trained using temporal difference error via backpropagation [3]. The TD error in (6) includes the actual reward r_{t+1}, the future rewards $Q(u_{t+1}, a_{t+1})$ that are expected to be obtained and the expected reward of the state-action pair $Q(u_t, a_t)$:

$$\delta_t = r_{t+1} + \gamma Q(u_{t+1}, a_{t+1}) - Q(u_t, a_t) \tag{6}$$

$$\delta_t^\lambda = \delta_t + \sum_{n=1}^{T-1} (\gamma\lambda)^n \delta_{t+n}\,, \tag{7}$$

where λ is the eligibility trace-decay parameter and γ is the discount factor. A global error-based expansive function $g(\delta)$ is defined to enhance the learning when an unexpected reward comes, which is shown as

$$g(\delta_t^\lambda) = \begin{cases} \dfrac{\delta_t^\lambda}{1-\delta_t^\lambda+\epsilon}, & 0 \le \delta_t^\lambda \le 1 \\ \delta_t^\lambda\,, & \delta_t^\lambda < 0 \text{ or } \delta_t^\lambda > 1, \end{cases} \tag{8}$$

where ε = 1e-4, here, a small constant to eliminate the singularity when $\delta_t^\lambda = 1$.

If the distance of u_t from all the previous centers is larger than ξ_U, we assign a new kernel center to this input u_t with the weight $w_{k,t}$. If the distance is smaller than ξ_U, the centers remain unchanged, and u_t is quantized to the closest center m. We locally update the center's coefficient accordingly, as follows:

$$\begin{cases} w_{k,t} = \eta g(\delta_t^\lambda), & t\text{: new center index} \\ w_{k,m} = w_{k,m} + \eta g(\delta_t^\lambda), & m\text{: closest center index.} \end{cases} \tag{9}$$

III. Results

First, we will visualize the simulated neural data structure obtained by principal component analysis (PCA) and show the decoding results and detailed analysis of the three models respectively.

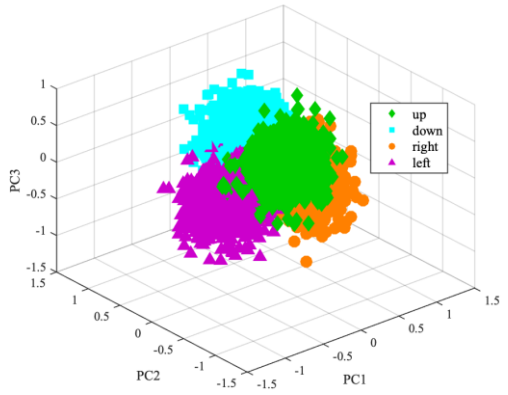

Figure 3. Neural patterns of four actions visualized by three main components of PCA.

In Fig. 3, we generate the spike data of four actions based on the tuning function in (1). Then we visualize the neural structure with the top three main components of PCA. We can see that the four clusters are separable, though with some overlaps, which imitates the real M1 neural data.

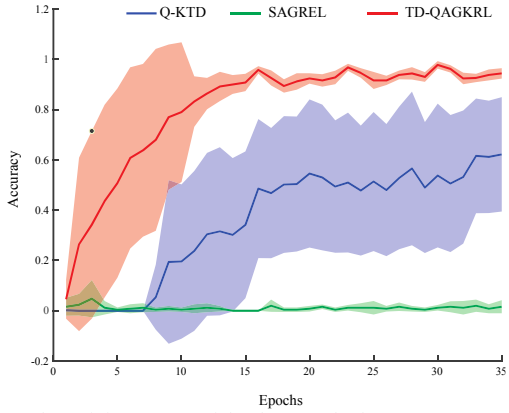

Figure 4. The training curves of the three methods.

We design the experiment with two steps to reach the peripheral targets and two steps to go back to the center, as described in Fig. 1. In other words, the subjects need four steps to finish one trial to get an external reward. We generate input data for the four respective directions and combine them in one trial. We shuffle 10000 trials for each initialization (70% for training, 30% for testing) and repeat the process 10 times. The input is the simulated neural data of the designed trajectory, and the output is the action sequence. The best parameters are explored for the three algorithms separately. We set the hidden number =10, discount factor $\gamma_1 = 0.99$, and learning rate $\eta_1 = 0.01$ for SAGREL; kernel size $\sigma_2 = 3$, learning rate $\eta_2 = 0.01$, quantization threshold $\xi_2 = 0.2$, discount factor $\gamma_2 = 0.99$ and temperature parameter $\tau_2 = 0.1$ for TD-QAGKRL; and kernel size $\sigma_3 = 3.5$, learning rate

$\eta_3 = 0.01$, quantization threshold $\xi_3 = 0.2$, discount factor $\gamma_3 = 0.99$ and $\varepsilon = 0.05$ for Q-KTD. The learning curves of the three methods are shown in Fig. 4. The x-axis is the training epoch (each contains 50 trials), and the y-axis is the decoding accuracy. The red, green and blue shaded curves (solid line shows mean value, and shaded area stands for the variance) represent TD-QAGKRL, SAGREL and Q-KTD, respectively. We can see that SAGREL cannot converge due to its local structure, while Q-KTD obtains a high performance but sometimes is trapped in the local minima when assigning credit over time due to its action-selection policy. It chooses an action with the maximal reward with a large possibility, and hardly explores the other actions, which may be better in the later training sessions. In comparison, TD-QAGKRL converges faster and has a higher convergence rate than Q-KTD, which is mainly contributed by the softmax policy and the expansive function.

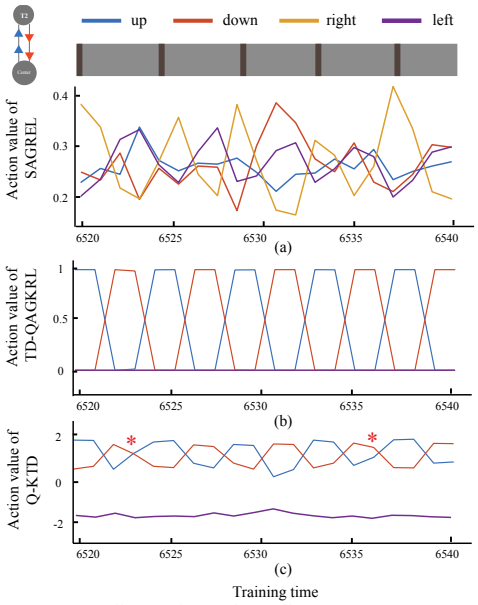

Figure 5. Corresponding action values of SAGREL, TD-QAGKRL and Q-KTD in the later training session.

Fig. 5 shows the action values of the four actions for the three methods in the later training session (which includes 5 trials, going up and then down). The x-axis is the training time, and y-axis is the action value. The brown stripes on the grey line indicate the start of each trial. As we can see from Fig. 5(a), SAGREL cannot distinguish between four actions after long-term training, while Fig. 5(b) shows that TD-QAGKRL can perfectly discriminate these actions and choose the correct action to complete the task. Q-KTD, meanwhile, can differentiate the actions well for most of the trials when it has a good initialization. But, compared with TD-QAGKRL, the actions are less discriminated, as indicated by the red stars in Fig. 5(c), and thus Q-KTD is highly likely to choose the wrong action. With the well-trained parameters, we test three algorithms on the data that never appeared in the training session. We conduct the testing on ten initializations (each containing 3000 points for testing). TD-GAGKRL (96.2% ± 0.77%) performs statistically better than Q-KTD (57.4% ± 40.5%) (pair-wise Student's *t*-test, $p<0.05$).

IV. Conclusion and Discussion

Reinforcement learning-based decoders have been utilized in BMIs to enable disabled people to control external devices, guided by a reward. However, when the task becomes more complex, it is difficult and time-consuming to train the decoder with the reward by the end of the trial. Here, we propose to embed temporal difference into the QAGKRL algorithm to ensure its performance. Compared with existing methods, our TD-QAGKRL algorithm can achieve better performance (96.2% ± 0.77%) and converges faster, with a smaller variance on simulated neural data. In future work, we plan to test TD-QAGKRL on *in vivo* data and extend this work in an online manner for clinical applications.

References

[1] M. A. Lebedev and M. A. L. Nicolelis, "Brain–machine interfaces: past, present and future," *Trends Neurosci.*, vol. 29, no. 9, pp. 536–546, Sep. 2006.

[2] J. Wessberg *et al.*, "Real-time prediction of hand trajectory by ensembles of cortical neurons in primates," *Nature*, vol. 408, no. 1, pp. 361–365, 2000.

[3] J. DiGiovanna, B. Mahmoudi, J. Fortes, J. C. Principe, and J. C. Sanchez, "Coadaptive brain-machine interface via reinforcement learning," *IEEE Trans. Biomed. Eng.*, vol. 56, no. 1, pp. 54–64, Jan. 2009.

[4] Y. Wang, F. Wang, K. Xu, Q. Zhang, S. Zhang, and X. Zheng, "Neural control of a tracking task via attention-gated reinforcement learning for brain-machine interfaces," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 23, no. 3, pp. 458–467, 2015.

[5] F. Wang *et al.*, "Quantized Attention-Gated Kernel Reinforcement Learning for Brain-Machine Interface Decoding," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 28, no. 4, pp. 873–886, 2017.

[6] X. Zhang, C. Libedinsky, R. So, J. C. Principe, and Y. Wang, "Clustering Neural Patterns in Kernel Reinforcement Learning Assists Fast Brain Control in Brain-Machine Interfaces," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 27, no. 9, pp. 1684–1694, Sep. 2019.

[7] T. Brosch, F. Schwenker, and H. Neumann, "Attention-gated reinforcement learning in neural networks - A unified view," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013, vol. 8131 LNCS, pp. 272–279.

[8] R. Sutton, "Introduction to Reinforcement Learning R A I L &."

[9] X. Shen, X. Zhang, Y. Huang, S. Chen, and Y. Wang, "Reinforcement Learning based Decoding Using Internal Reward for Time Delayed Task in Brain Machine Interfaces," in *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, 2020, vol. 2020-July, pp. 3351–3354.

[10] J. Bae, L. G. Sanchez Giraldo, E. A. Pohlmeyer, J. T. Francis, J. C. Sanchez, and J. C. Príncipe, "Kernel temporal differences for neural decoding," *Comput. Intell. Neurosci.*, vol. 2015, 2015.

[11] J. Bae, P. Chhatbar, J. T. Francis, J. C. Sanchez, and J. C. Principe, "Reinforcement learning via kernel temporal difference," in *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, 2011, pp. 5662–5665.

[12] C. Libedinsky *et al.*, "Independent Mobility Achieved through a Wireless Brain-Machine Interface," *PLoS One*, vol. 11, no. 11, p. e0165773, Nov. 2016.

[13] B. Amirikian and A. P. Georgopulos, "Directional tuning profiles of motor cortical cells," *Neurosci. Res.*, vol. 36, no. 1, pp. 73–79, Jan. 2000.