NON-LOCAL SELF-ATTENTION STRUCTURE FOR FUNCTION APPROXIMATION IN DEEP REINFORCEMENT LEARNING

Zhixiang Wang[1], *Xi Xiao*[1], *Guangwu Hu*[2*], *Yao Yao*[1]
Dianyan Zhang[1], *Zhendong Peng*[1], *Qing Li*[3], *Shutao Xia*[1,3]

[1] Tsinghua University, Beijing, China
[2] School of Computer Science, Shenzhen Institute of Information Technology, Shenzhen, China
[3] Southern University of Science and Technology, Pengcheng Laboratory, Shenzhen, China

ABSTRACT

Reinforcement learning is a framework to make sequential decisions. The combination with deep neural networks further improves the ability of this framework. Convolutional nerual networks make it possible to make sequential decisions based on raw pixels information directly and make reinforcement learning achieve satisfying performances in series of tasks. However, convolutional neural networks still have own limitations in representing geometric patterns and long-term dependencies that occur consistently in state inputs. To tackle with the limitation, we propose the self-attention architecture to augment the original network. It provides a better balance between ability to model long-range dependencies and computational efficiency. Experiments on Atari games illustrate that self-attention structure is significantly effective for function approximation in deep reinforcement learning.

Index Terms— reinforcement learning, deep learning, self-attention, Atari game

1. INTRODUCTION

Recently, reinforcement learning has achieved satisfying performance in many tasks such as video games[1, 2] and chess game Go[3, 4]. It is a framework to make sequential decisions. In the formulation of reinforcement learning, an agent interacts with an environment over time. At timestep t, the agent observes the state s_t defined in a state space S and selects an action a_t defined in an action space A. The action chosen is decided by a policy $\pi(a_t|s_t)$. Then the agent receives an instant reward r_t and makes a transition to next state s_{t+1} following reward function $R(s,a)$ and transition probability $P(s_{t+1}|s_t,a_t)$ respectively. The $R(s,a)$ and $P(s_{t+1}|s_t,a_t)$ are decided by the environment dynamics. Generally, the agent repeats this process until it reaches a terminal state. The final return R_t is defined as the discounted accumulation of sequential instant rewards $\sum_{k=0}^{\infty}\gamma^k r_{t+k}$ with a discount factor $\gamma \in (0,1]$. During the process, the agent adjusts the policy $\pi(a_t|s_t)$ to maximize $E_s\left[\sum_{k=0}^{\infty}\gamma^k r_{t+k}\right], s \in S$.

Various game environments are adopted to compare different reinforcement learning algorithms (e.g. AC [5], A3C [6], TRPO [7], PPO [8], ACKTR [9], Deep Q-Network(DQN) [2]) which are ranked based on the final score in certain game environments. The Arcade Learning Environment(ALE)[1] is composed of more than 50 Atari games. It provides raw pixels for agents to learn a policy. In ALE[1], DQN[2] was the first learning algorithm that showed human expert-level control. A convolutional neural network was used to approximate $Q(s,a)$ in the framework of Q-learning. For games with partially observed states, Deep Recurrent Q-network adds a recurrent neural network layer before output and achieves satisfying performance. In[10], deep deterministic policy gradient is proposed by training a CNN-based agent end-to-end in TORCS[11]. [12] demonstrated that a CNN-based agent can achieve human-like behaviors in basic scenarios in a first-person shooter game environment.

From the perspective of deep learning, reinforcement learning faces a few challenges. Firstly, Deep learning requires large amounts of labelled data while the scalar reward in reinforcement learning is generally sparse, noisy and delayed. Additionally, Deep learning assumes data samples to be independent and follow a fixed distribution, while states in reinforcement learning are generally highly correlated with a consistently changing distribution. V Mnih et al.[1, 2] demonstrated that a convolutional neural network can overcome these problems to learn a human-level policy from raw pixels directly. However convolutional neural networks still have own limitations. The ability to capture geometric or structural patterns may be not satisfying. For example, in CNN-based image generative models[13], the texture information is usually generated well and the geometric information is generally ignored. The generative model creates realistic image classes such as sky, ocean and landscape, while for other image classes such as dog or cat, it generates realistic fur texture and fails to create clearly defined separate

*Corresponding Author: Guangwu Hu(hugw@sziit.edu.cn)

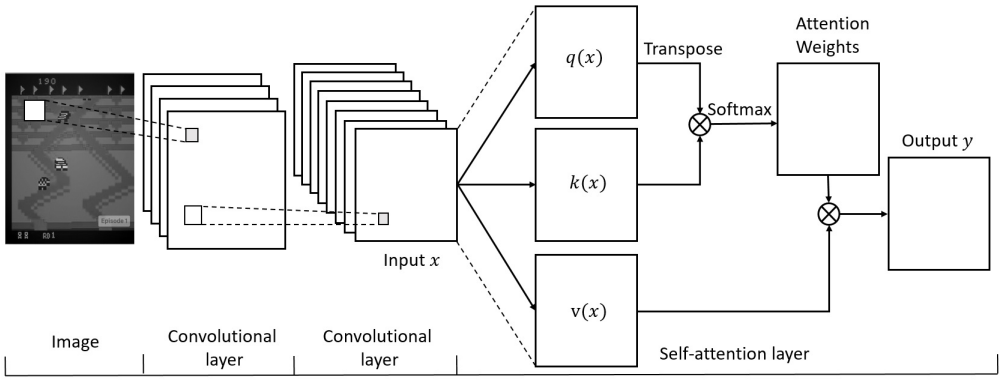

Fig. 1. Structure of a self-attention layer. $\otimes$ means matrix multiplication. The Softmax is performed on each row inside one channel. $q(x), k(x), v(x)$ are implemented as 1×1 convolutions.

feet in the right place. one possibly reasonable explanation is that different image regions can only build up dependencies and relationships through convolution operation with a local receptive field. Long-range dependencies can only be built up through a enough large number of convolutional layers. Increasing the kernel size of convolutional operation may help to connect different image regions better but will also lose the computational efficiency obtained by local convolutional operation.

In reinforcement learning problem setting, geometric information and dependencies over different image regions may be generally more important than that in other tasks such as image classification and image generation. The recently proposed self-attention structure[14, 15] is a better choice to model long-range dependencies among different image regions in a computationally efficient way. The self-attention mechanism calculates the response at a position as a weighted sum of the features at all other positions with only a small computational cost. The self-attention structure has also been adopted in other tasks such as generative adversarial network[16] and video classification[17].

In order to tackle with the weakness of convolutional networks in build up long-range dependencies, we introduce the self-attention structure into the reinforcement learning framework. For function approximation, the self-attention structure helps the original convolutional neural network to strengthen the ability to process long-range dependencies over different image regions which contains more geometric information. It perform computations among different regions of feature maps so that relationships among various regions can be formulated during the training process. To evaluate our proposed method, we conduct experiments on Atari tasks defined in OpenAI Gym[18]. The Adoption of this structure is shown to be an effective way to balance long-range dependencies and computational efficiency. To the best of our knowledge, it is the first time to adopt self-attention structure into function approximation into reinforcement learning.

2. SOLUTION DESCRIPTION

2.1. Reinforcement Learning Formulation

In reinforcement learning, the policy $\pi(a|s)$ was represented by a machine learning module. In deep reinforcement learning, this machine learning module is a neural network. State value function $V(s) = E[\sum_{k=0}^{\infty} \gamma^k r_k | s]$ and action value function $Q(s,a) = E[\sum_{k=0}^{\infty} \gamma^k r_k | s, a]$ are also represented by machine learning modules (e.g. convolutional neural network) to guide the policy to update itself towards a right direction. Functions $\pi(a|s), V(s)$ and $Q(s,a)$ need to be approximated by proper machine learning modules. Different representation abilities of modules lead to different final performances.

Table 1. Final scores of 45 Atari games. The score is computed by taking average of $Rewards$ over final 10000 timesteps.

Game	CNN	Self-attention	Game	CNN	Self-attention	Game	CNN	Self-attention
Amidar	275	**391**	Frostbite	250	**251**	PrivateEye	64.8	**144**
Asterix	5588	**6510**	Gopher	907	**924**	Qbert	12412	**14142**
Atlantis	1729268	**1958670**	Gravitar	335	**372**	Riverraid	7426	**9100**
BankHeist	1128	**1173**	Hero	19462	**19957**	RoadRunner	32470	**32984**
BeamRider	4184	**4252**	IceHockey	-6.8	**-6.5**	RoboTank	2.2	**2.8**
Berzerk	766	**785**	JamesBond	**2429**	2312	Seaquest	1701	**1727**
Bowling	23.7	**24.1**	Kangaroo	**559**	462	Skilling	-28680	**-16189**
Breakout	382	**387**	Krull	7886	**7903**	SpaceInvader	758	**803**
Centipede	**4178**	3812	KungFuMaster	**28564**	27334	StarGunner	44623	**45296**
Chopper Command	995	**1362**	Montezuma's Revenge	0.1187	**0.1209**	TimePilot	**3491**	3431
CrazyClimer	103959	**105340**	MsPacMan	1942	**2127**	Tutankhan	176	**185**
DemonAttack	18005	**22932**	NameThisGame	5658	**5750**	UpNDown	31341	**43661**
DoubleDunk	**-15.8**	-16.1	Phoenix	13188	**14938**	Venture	0	0
FishingDerby	**21.5**	16.6	Pitfall	-65.2	**-50.3**	YarsRevenge	**13793**	12365
Freeway	0.0017	0.0017	Pong	18.1	**19.5**	Zaxxon	15.3	**21.4**

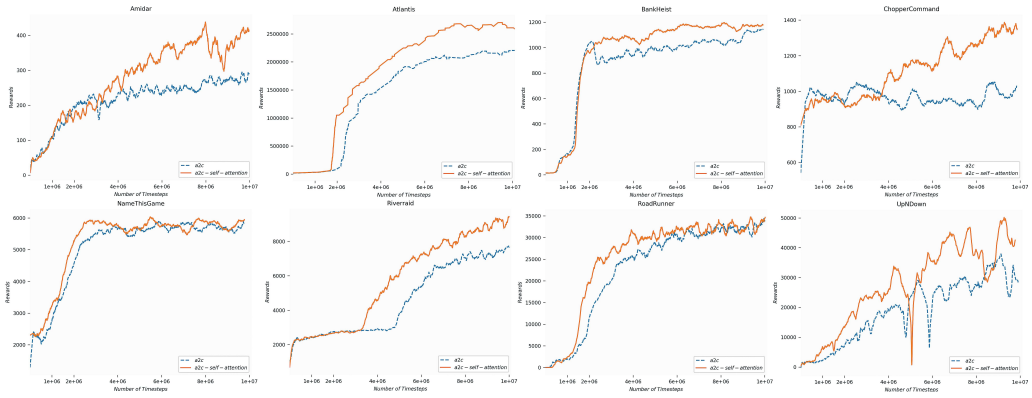

Fig. 2. Training process of 8 games. $Rewards$ means final episode return in one round of a game. It is computed by summing up all instant rewards of every timestep in that round of the game. The agent gets a state input and output an action in one timestep.

2.2. Self-Attention Function Approximation

For CNN-based function approximation, the input raw pixel states are processed in a local receptive field. The structure of convolutional layers decides that it is an computationally inefficient way to model long-range dependencies between widely separated spatial regions. The proposed structure[17] is adopted into deep reinforcement learning framework to extract features of input states.

The input pixel image is first processed by several convolutional layers and transformed into image features $x \in R^{C\times H\times W}$ where C means 'channel', H means 'height' and W means 'width'. x_i represents the element in ith position among all the $C \times H \times W$ positions. The input x is duplicated into 3 copies which are query x_{query}, key x_{key} and value x_{value}[15]. Then we do transformation and get $q(x_{query}) = W_{query}x_{query}$, $k(x_{key}) = W_{key}x_{key}$ and $v(x_{value}) = W_{value}x_{value}$ seperately. The metric W_{query}, $W_{key}$and W_{value} are the network parameters to be learned. Since x_{query}, x_{key}, x_{value} are the same, we just use $q(x)$, $k(x)$, $v(x)$ to represent $q(x_{query})$, $k(x_{key})$, $v(x_{value})$ and $s_{ij} = q(x_i)\, k(x_j)$. The output of the self-attention layer has the same size as the input. When we rebuild the element in the jth position, $a_{j,i}$ indicates how much attention should be paid to the ith location.

$$a_{j,i} = \frac{\exp(s_{ij})}{\sum_{j=1}^{C\times H\times W} \exp(s_{ij})}$$

The output element in the jth position is calculated in the following way.

$$y_j = \sum_{i=1}^{C\times H\times W} a_{j,i} v(x_i)$$

Finally, we multiply the output by a parameterized scalar β and add it back to the input feature map. Then we get the final output y_{out}.

$$y_{out} = \beta y + x$$

3. EXPERIMENT SETTING

To evaluate our method, we conduct experiments on the Atari game tasks defined in OpenAI Gym[18] with A2C algorithm which is a synchronized version of A3C algorithm[10].

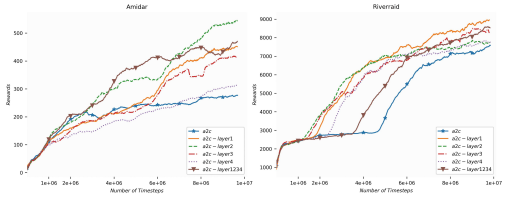

Fig. 3. Comparison of different positions to add the self-attention layer. $layer1$ means adding it before the first convolutional layer. $layer1234$ means adding four self-attention layers.

We adopt the same proprecess as [1, 2]. Raw Atari frames are 210×160 pixel images with a 128 color palette. For computational efficiency, we convert the RGB image to a gray-scale image and then down-sample it to 110×84. Finally we crop it and get an 84×84 region which captures the playing area. We stack 4 84×84 images together as the state input.

The state value function and the policy share one network to extract features. The first layer is a convolutional layer composed of 32 8×8 filters and the stride is 4. The second convolutional layer consists of 64 4×4 filters and the stride is 2. The third convolutional layer contains 32 3×3 filters and the stride is 1. The following is a fully connected layer with a 512-dimensional output. In the last layer, we get our action output and state value.

We conduct following experiments. To show the effectiveness of the self-attention structure, we add a self-attention

Fig. 4. Comparision of different structures. The self-attention layer is added before the fully connected layer. $Conv1$ means replacing it with a convolutional layer composed of 32 1×1 filters. $Conv2$ means replacing it with a convolutional layer composed of 32 3×3 filters. The stride is 1.

layer to the original convolutional network and test the final scores on 45 Atari games. Then we replace the self-attention layer with two kinds of convolutional layers to do comparison. Finally, we try to add the self-attention layer to different positions of the network and compare the difference. Practically, we simplify $q(x)$, $k(x)$, $v(x)$ as 1×1 convolutions of feature maps with stride 1 and softmax is performed only on each row inside each feature channel.

4. EXPERIMENT RESULT

To show the effectiveness of a self-attention layer, we add it between the third convolutional layer and the fully connected layer. We conduct experiments on 45 Atari Games based on OpenAI Gym[18] environment. The results are shown in Table 1. As we can see, most games gain benefit and perform better. Among all these 45 games, about 15 ones gain significant improvement. We choose 8 of them randomly and conduct further experiments to do analysis. The training processes of the 8 games are shown in Figure 2, and we can see the improvement in terms of final episode returns and training efficiency. The 'Rewards' in the figure means final episode return in one round of a game. We sum up all the instant rewards of every timestep in one round of a game to get the final episode return. In one timestep, the agent observes the state once and choose an action. In a self-attention layer, input and output has the same number of channels.

To exclude the influence brought by more parameters, we replace the self-attention layer with a convolutional layer composed of 32 1×1 filters with the stride 1. This structure has the same number of parameters as the self-attention layer between the third convolutional layer and fully connected layer. what is different is the way to utilize these parameters. Generally, a convolutional layer with more parameters means better feature extraction ability. To show the effectiveness brought by our self-attention structure further, we also replace the self-attention layer with another convolutional layer composed of 32 3×3 filters and the stride is 1. This convolutional layer needs more parameters than the self-attention layer. If it performs worse than the self-attention layer, we may be more confident to contribute the performance improvement to the non-local self-attention structure. Inevitably, in this setting, the fully connected layer will change from original '$32 \times 7 \times 7$' $\times$ 512 to '$32 \times 5 \times 5$' $\times$ 512. As shown in Figure 4, we can see, generally, self-attention performs better than the other two settings. Comparing the same games in Figure 2 and Figure 4, it is shown that more parameters don't always lead to better performance. In several games, more parameters and improper structure lead to worse performance.

To explore a better way to utilize self-attention structure and how a self-attention layer brings effect, we employ the self-attention layer in different positions, before the first convolutional layer, before the second convolutional layer, before the third convolutional layer, before the forth fully connected layer and before all these layers with four self-attention layers. As shown in Figure 3, we can see, self-attention layers provide benefit on each position and more self-attention layers may not always perform best.

5. CONCLUSIONS

In order to tackle with the weakness of convolutional networks in building up long-range dependencies, in this paper, we incorporate the self-attention mechanism into the deep reinforcement learning framework and propose the self-attention function approximation. This structure can build up relationships among different regions of the state input effectively with satisfying computational efficiency. Our experiments illustrate the effectiveness of the self-attention structure compared with a pure convolutional structure.

6. ACKNOWLEDGMENTS

This work is supported by the NSFC projects(61773229, 61771273, 61202358), the National High-tech R&D Program of China(2015AA016102), Guangdong Natural Science Foundation(2018A030313422), the RD Program of Shenzhen(JCYJ20160531174259309, JCYJ20170307153032483, JCYJ20160331184440545, JCYJ20170307153157440, JCYJ 20170817115335418).

7. REFERENCES

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller, "Playing atari with deep reinforcement learning," *CoRR*, vol. abs/1312.5602, 2013.

[2] V Mnih, K Kavukcuoglu, D Silver, A. A. Rusu, J Veness, M. G. Bellemare, A Graves, M Riedmiller, A. K. Fidjeland, and G Ostrovski, "Human-level control through deep reinforcement learning.," *Nature*, vol. 518, no. 7540, pp. 529, 2015.

[3] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

[4] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, pp. 354–, Oct. 2017.

[5] R. S. Sutton, "Policy gradient methods for reinforcement learning with function approximation," *Advances in Neural Information Processing Systems*, vol. 12, pp. 1057–1063, 2000.

[6] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, 2016, pp. 1928–1937.

[7] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel, "Trust region policy optimization," *Computer Science*, pp. 1889–1897, 2015.

[8] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *ArXiv e-prints*, July 2017.

[9] Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba, "Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., pp. 5279–5288. Curran Associates, Inc., 2017.

[10] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra, "Continuous control with deep reinforcement learning," *Computer Science*, vol. 8, no. 6, pp. A187, 2015.

[11] Bernhard Wymann, Christos Dimitrakakisy, Andrew Sumnery, and Christophe Guionneauz, "Torcs: The open racing car simulator," 2015.

[12] Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski, "ViZDoom: A Doom-based AI research platform for visual reinforcement learning," in *IEEE Conference on Computational Intelligence and Games*, Santorini, Greece, Sep 2016.

[13] Takeru Miyato and Masanori Koyama, "cGANs with projection discriminator," in *International Conference on Learning Representations*, 2018.

[14] Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit, "A decomposable attention model for natural language inference," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November*, 2016.

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *NIPS*, 2017.

[16] Han Zhang, Ian J. Goodfellow, Dimitris N. Metaxas, and Augustus Odena, "Self-attention generative adversarial networks," *CoRR*, vol. abs/1805.08318, 2018.

[17] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He, "Non-local neural networks," in *CVPR*, 2018.

[18] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba, "Openai gym," *CoRR*, vol. abs/1606.01540, 2016.