Memory-Augmented Episodic Value Network

Fanyu Zeng, *Member, IEEE*, Guangyu Xing, and Guang Han

***Abstract*—In this paper, we propose memory-augmented episodic value network (M-EVN) to learn a differentiable planning-based policy with episodic memory in maze games. The episodic memory module associates the environmental state to its corresponding state value function, and outputs a weighted sum of state value functions with similar states to improve the agents' navigation performance in partially observable mazes. In addition, we introduce a Net-in-Net architecture to make M-EVN differentiable by error backpropagation and learn an explicit planning computation. We train M-EVN in 2D maze games, and the experimental results show that the M-EVN agent outperforms the original value iteration network (VIN) in the partially observable maze games.**

***Index Terms*—Episodic Memory, Value Network, Deep Reinforcement Learning, Visual Navigation.**

I. Introduction

DEEP reinforcement learning (DRL) [1] is widely used in video games, robot operation, agent navigation and other fields [2], [3], [4]. Most DRL works based on neural network structures [5], [6], [7] adopt the structure of convolutional layers and fully connected layers, in which the convolutional layers are used to extract the features of the environment state, and the fully connected layer maps the environmental features into the probability distribution of the actions, and finally trains a mapping relationship between the state and the actions, namely the policy. The learning method of aforementioned neural networks reduces the loss function by continuously selecting the suitable actions for navigation agent to obtain good policy. However, such navigation policy is difficult to generalize to new environments, hence that policy is called reactive policy [8]. Reinforcement learning solves markov decision problems, and essentially solves a sequential decision problem where the influence of subsequent decisions on the current decision needs to be considered. Nevertheless, the reactive policy does not express the impact of subsequent decisions on the current decision. Therefore, the reaction formula lacks planning ability [8].

To address the issue of the lack of planning ability of the reactive policy, Tamar et al. [8] embedded a planning module into a differentiable neural network, called value iteration network (VIN). VIN is a differentiable value iterative approximation algorithm, and its computational process can be regarded as a convolutional neural networks [9] based on end-to-end error backpropagation [10]. In addition, the reason why human beings can adapt to complex environments is that they own memory [11], and memory is very important for agents to realize more advanced behaviors [12]. Introducing memory into the deep reinforcement learning model can make full use of prior knowledge, and make agents directly utilize the information collected from past events to guide their behavior. Moreover, memory enables agents to have certain degree of active cognition and reasoning ability, and helps agents navigate in the complex environments better.

In recent years, researchers have proposed several memory-augmented neural networks (MANNs), including memory network [13], [14], episodic memory [15] and differentiable neural computer [16]. Compared with the other two MANNs, episodic memory has several advantages [17]: (1) improving the efficiency of agent's samples in complex state space; (2) learning based on small amount of samples; (3) effective approximation of state value function; (4) establishing long-term dependence between actions and rewards. Episodic memory stores past observations in episodic memory and absorbs new experiences into future actions. Specifically, environmental features and the past state values are stored in episodic memory, and the value function estimation of the current state is calculated by combining the values of similar states. [17], [18]. To improve agent's navigation performance in partially observable environments, episodic memory is introduced into the value iteration network to enhance the agent's navigation ability.

In this paper, we improve the agent's navigation performance in partially observable maze games, and a reinforcement learning navigation augmented with episodic memory is proposed. First, episodic memory is introduced into value iteration network, and its mathematical model is analyzed in details. In addition, the calculation process of episodic memory is regarded as the convolutional computation process. Then, a Net-in-Net architecture is constructed, and the proposed algorithm updates parameters using error backpropagation. Finally, the experimental results show that compared with the value iteration network without episodic memory, the memory-augmented epiosidc value network performs better in the same partially observable maze.

A memory-augmented episodic value network (M-EVN) is proposed in this work, and its contributions are summarized as follows:

(1) For better navigation performance in a partially observable maze games, we store the past encoded states and the corresponding values into episodic memory. Then, calculating the approximate state value function based on its similar state estimations.

(2) A Net-in-Net architecture, which can be regards as a convolutional computation, is proposed to make M-EVN

This work was supported by National Natural Science Foundation of China (U1813202, 61773093, 62003381, 62172290, 61871445); National Key R&D Program of China (2020YFB1313600 ,2018YFC0831800); and NUPTSF (NY221040).

Fanyu Zeng and Guang Han are with the Engineering Research Center of Wideband Wireless Communication Technology, Ministry of Education, Nanjing University of Posts and Telecommunications, Nanjing, China (e-mail: zengfanyu@njupt.edu.com).

Guangyu Xing is with the College of Computer Science, Sichuan University, Chengdu, China.

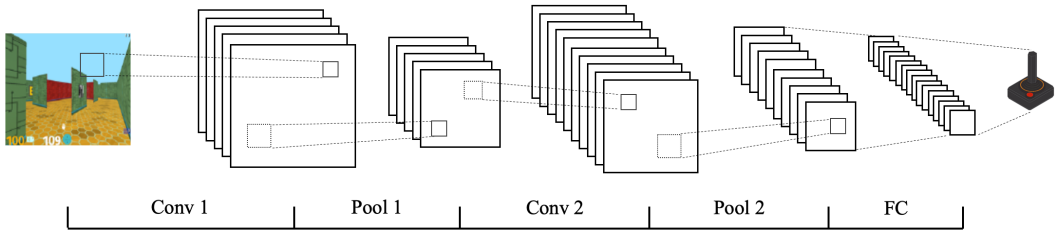


Fig. 1: The basic structure of convolutional neural networks, which include convolutional layers, pooling layers, and fully connected layer.

differentiable and have planning ability.

The remainder of this paper is organized as follows: Section II presents the related works, including convolutional neural networks, value iteration network, and episodic memory. Then, Section III describes memory-augmented episodic value network in details. In Section IV, the experimental results are presented and discussed, which shows the advantage of M-EVN. Finally, the conclusions are summarized in Section V.

II. RELATED WORKS

A. Convolutional Neural Networks

In recent years, deep learning [9], [19], [20] have been used in supervised learning [21], [22], unsupervised learning [23] and reinforcement learning [3], [7]. The most-used neural network architecture of deep learning is convolutional neural networks. Convolutional neural networks are widely used in various deep learning algorithms, such as AlexNet [24], VGG [25], Googlenet [26], ResNet [27], DesNet [28], MobileNet [29], ShuffleNet [30] and GAN [31].

As shown in Fig. 1, the basic structure of convolutional neural networks include convolutional layer, pooling layer and fully connected layer. The convolutional layer convolves the input image and extracts its image features, the pooling layer conducts subsample operation on image features to reduce their dimensions, and the fully connected layer finally classifies the learned features. Fig. 1 contains two convolutional layers, two pooling layers and one fully connected layer. The convolutional layers and the pooling layers work together to learn convolved image features, and the fully connected layer outputs the probability of classification/decision.

The convolutional layer is the core of convolutional neural networks [9], [32], which include an amount of different convolutional cores, and is used to learn the local features of images. The pooling layer is connected to the convolution layer, and it subsamples the convolved feature maps. Moreover, the frequently-used pooling layes include average pooling layer and max pooling layer. The fully connected layer is connected to multiple convolutional layers and pool layers, and each of its neurons is connected with all the neurons in the previous layer. The fully connected layer acts as a classifier which maps the feature vectors outputed by the pooling layer to actions' probability, and the action with the maximum probability corresponds to the current optimal action.

Assuming that the size of input image is $m \times n$ and its channel is l, the output of the convolutional layer has l' channels and its convolutional kernel is $(W^1, W^2, ..., W^{l'})$. After the input images convolved by the convolutional layer, the feature maps outputed can be expressed as follows [8]:

$$h_{l',i',j'} = \sigma(\textstyle\sum_{l,i,j} W^{l'}_{l,i,j} X_{l,i'-i,j'-j}) \tag{1}$$

where σ is the activation function.

Then, the feature maps of the convolutional layer is fed into the pooling layer, and the pooling layer outputs as follows [8]:

$$h^{maxpool}_{l,i,j} = \max_{i',j' \in N(i,j)} h_{l,i',j'} \tag{2}$$

where $N(i,j)$ is the k neighborhoods around pixel (i,j).

Finally, the feature maps from the pooling layer is fed into the fully connected layer for the probability distribution of diferent actions, and the action corresponding to the maximum probability is chosen as the policy output.

B. Value Iteration Network

Tamar et al.[8] introduced value iteration network, which is a fully differentiable neural network with a 'planning module' embedded within. In Fig. 2, the VIN reframes value iteration as the computation of convolutional neural network, and regards Q function as each channel in convolutional layer, and its discounted probability corresponds to the weight of convolutional kernel. The equation of VIN is expressed as follows [8]:

$$\begin{aligned} \bar{Q}^{(k)}_{\bar{a},i',j'} &= \sum_{i,j} (W^R_{\bar{a},i,j} \bar{R}_{i'-i,j'-j} + W^V_{\bar{a},i,j} \bar{V}^{(k-1)}_{i'-i,j'-j}) \\ \bar{V}^{(k)}_{i,j} &= \max_{\bar{a}} \bar{Q}^{(k)}_{\bar{a},i,j} \end{aligned} \tag{3}$$

where $i,j \in [m]$ corresponds to cells in the $m \times m$ maze, $\bar{R}$, $\bar{Q}$, $\bar{V}$ is the estimated reward, action value function and state value function, respectively. $\bar{a}$ is the action index of the feature map $\bar{Q}$. W^R and W^V are the convolutional weights for the reward function and state value function, respectively. In the next iteration, the previous value function $\bar{V}$ stacked with the current reward $\bar{R}$ is fed into the convolutional layer and max pooling layer for K iterations, as shown in Fig. 2.

VIN regards each iteration of VI module as passing the previous state value function $\bar{V}$ and reward function $\bar{R}$ into convolutional layers and max pooling layers. Hence, VIN is differentiable and can be updated using error backpropagation algorithm.

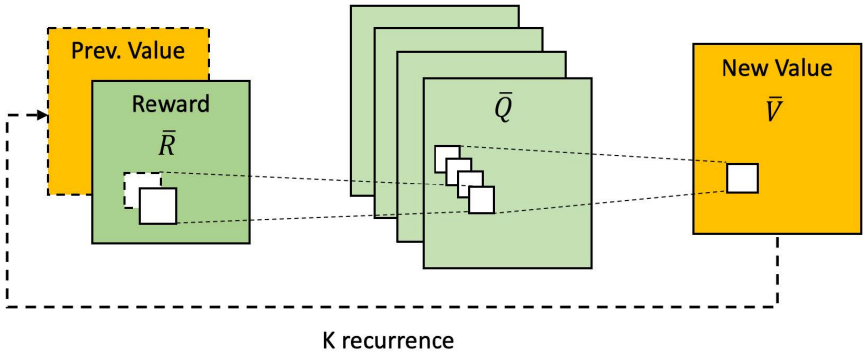


Fig. 2: The structure of VI. VI is regarded as the computation of convolutional neural networks.

C. Episodic Memory

Episodic memory [17], [33] refers to the memory of an individual at a specific time and place. The classic structure of episodic memory is differentiable neural dictionary (DND) [15]. As shown in Fig. 3, DND uses a semi-table representation of value function to store n simple memory units $M_i = (K_i, V_i)$, the key K_i and its corresponding value V_i are correlated, and they have the same number of vectors.

Key	Value
K_1	V_1
K_2	V_2
⋮	⋮
K_n	V_n

Fig. 3: The structure of DND. The DND has a memory module (keys, values), where keys and values are dynamically sized arrays of vectors, each containing the same number of vectors.

The DND is a buffer pool of past experiences with slowly changing state representations, and can rapidly update value function estimates. The advantages of episodic memory are as follows [15], [17]:
(1) improving the efficiency of training data;
(2) approximating better value function estimation in a complex state space;
(3) learning based on small amount of training data;
(4) establishing long-term dependence between action and reward.

DND can provide better approximation for the value function in reinforcement learning. For each action $a \in \mathcal{A}$, $M_a = (K_a, V_a)$ is stored in DND, where K_a and V_a are dynamically sized array of vectors, each containing the same number of vectors. DND associates keys with corresponding value and has two operations: lookup and write.

The lookup operation maps a key h to an output value o:

$$o = \sum_i w_i v_i \tag{4}$$

where v_i is the i-th element of array V_a, and w_i is the normalized parameter. The output w_i of a lookup in a DND is a weighted sum of the values in the memory, whose weights are given by normalized kernels between the lookup key and the corresponding key in memory.

$$w_i = k(h, h_i) / \sum_j k(h, h_j) \tag{5}$$

where h_i is the i-th element of array K_a, and $k(x, y)$ is gaussian or inverse kernels between vectors x and y.

Write operations store key-value pairs into memory. Moreover, if a key already exists in DND, only the corresponding value is updated.

III. Method

In this section, the proposed method are introduced. In III-A, the memory-augmented episodic value network is formulated. Then, in III-B, we present the neural network architecture of the M-EVN for agent navigation in details.

A. Problem Formulation

The structure of M-EVN is shown in Fig. 4, and the computation of M-EVN includes three steps: firstly, stacking the previous value function map and reward map to obtain the current action value function by policy evaluation, and then computing the new state value function. Secondly, multiple similar state values can be found by searching episodic memory. Finally, these similar state values are calculated through a Net-in-Net module to get the final optimal value function estimation.

Combining VIN with episodic memory, each state value function is estimated by similar states. In addition, a Net-in-Net architecture is proposed to make M-EVN differentiable. The policy evaluation equation is as follows:

$$\begin{aligned} \hat{Q}(s,a) &= R(s,a) + \gamma \textstyle\sum_{s'} P(s'|s,a) \hat{V}_M(s') \\ &= R(s,a) + \gamma \textstyle\sum_{s'} P(s'|s,a) \sum_{i=1}^{N} w_i(s') \hat{V}(i) \end{aligned} \tag{6}$$

where $R(s,a)$ is the reward, $P(s'|s,a)$ is the transition probability function that encodes the probability of the next state s' given the current state s and action a. $V_n(s')$ is the value function for the next state s', and γ is the discount ratio.

Given a memory module M and a state s', we refine the N most similar states $M_{s'} \subset M$ according to a distance metric $d(\bullet, s')$. Then a memory-based value estimation is calculated as follows:

$$\hat{V}_M(s') = \sum_{i=1}^{N} w_i(s') \hat{V}(i) \tag{7}$$

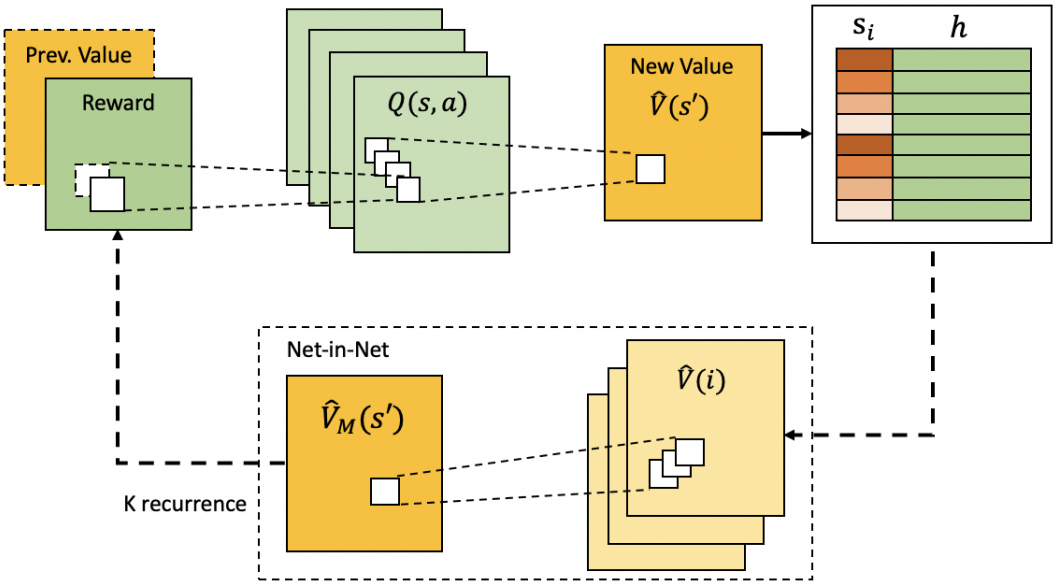


Fig. 4: The flowchart of M-EVN. M-EVN includes a Net-in-Net architecture.

where $\sum_{i=1}^{N} w_i(s') = 1$, and the distance metric $d(\bullet, s')$ uses euclidean distance. Eq. 7 is a Net-in-Net architecture which can be regarded as convolutional computation. As shown in Fig. 4, the Net-in-Net module has planning ability similar to the original VIN. In addition, the Net-in-Net module and the episodic memory are based on VIN, hence M-EVN can learn a differentiable planning-based policy.

According to the above analysis, policy evaluation of M-EVN can be written as follows:

$$\begin{aligned} \hat{Q}(s,a) &= R(s,a) + \gamma \textstyle\sum_{s'} P(s'|s,a)\hat{V}_M(s') \\ &= R(s,a) + \gamma \textstyle\sum_{s'} P(s'|s,a)\sum_{i=1}^{N} w_i(s')\hat{V}(i) \end{aligned} \tag{8}$$

Hence, the optimal value function estimation of M-EVN is as follows:

$$\begin{aligned} &\hat{V}_{n+1} = \max_a \hat{Q}(s,a) \\ &= \max_a (R(s,a) + \gamma \textstyle\sum_{s'} P(s'|s,a)\sum_{i=1}^{N} w_i(s')\hat{V}(i)) \end{aligned} \tag{9}$$

Then, using temporal difference (TD) to construct the loss function of M-EVN. The n-step TD loss function can be written as follows:

$$L(w) = E[(r + \max_{a'} \hat{Q}(s', a'w) - \hat{Q}(s,a,w)] \tag{10}$$

where r is the sum of n-step discounted reward, and w is the parameters of its neural network. s' and a' are the n-step state and action, respectively.

B. Neural Network Architecture

The neural network architecture used in M-EVN is shown in Fig. 5. The input images are 8×8 grayscale, and two continuous input images are convolved by the convolutional kernel with the size of 3×3 and the step size of 1 to obtain image features F. Then, the reward map R is obtained by the same convolution kernel convolved with F, and the reward map R and the memory-based value estimation $\hat{V}_M$ are stacked to be vector $[R, \hat{V}_M]$. Further, $[R, \hat{V}_M]$ is convolved with the same kernal to output the action value function map Q, and the state value function $\hat{V}$ is computed by max pooling.

According to the distance metric $d(\bullet, s')$, N most similar state value function maps $\hat{V}$ of the current state are searched from DND. Then $\hat{V}$ are convolved with the convolution kernel of size of 3×3 and step size of 1 to obtain the memory-based value function estimation $\hat{V}_M$. In addition, the environment gives the agent rewards as the feedback of its actions when the agent interacts with environment. Hence, the environment generates new reward map R. The new reward map stacks the value estimation based on episodic memory again to start a new iteration to improve the learning effectiveness.

IV. EXPERIMENTS

To verify the effectiveness of memory-augmented episodic value network in reinforcement learning navigation, VIN agent and M-EVN agent were used to conduct navigation experiments in 2D Gridworld [34] maze game. A certain number of training times were set for each maze game, and the successful number was recorded when the agent successfully reaches target object. In addition, the navigation performace of VIN agent and M-EVN agent were recorded, respectively.

A. Experimental Settings

We perform the experiments using TensorFlow, and train M-EVN on an Nvidia GeForce GTX 1080Ti GPU and an Intel Xeon E5-2696 v3@2.30GHz×36 CPU.

The hyperparameters used in M-EVN are shown in Table I. The training times are set to 2000, meaning that the agent trains 2000 times in each maze game. DND capacity is 500, which means that DND has 500 memory units, namely "key-value" pairs; The DND size is 100, which means the length

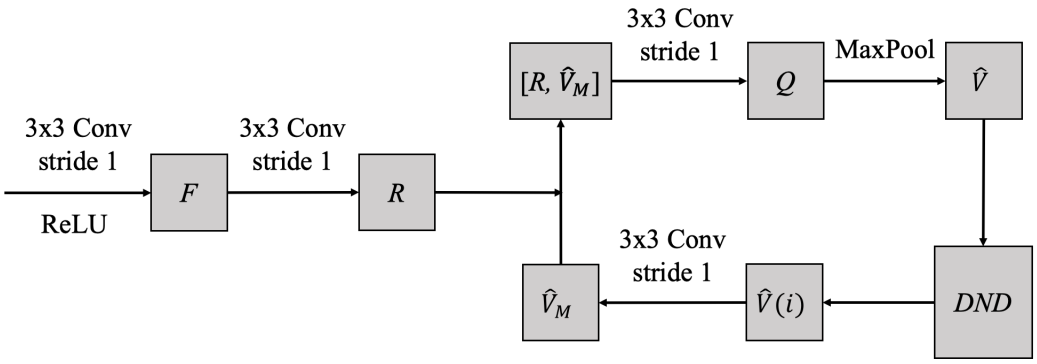
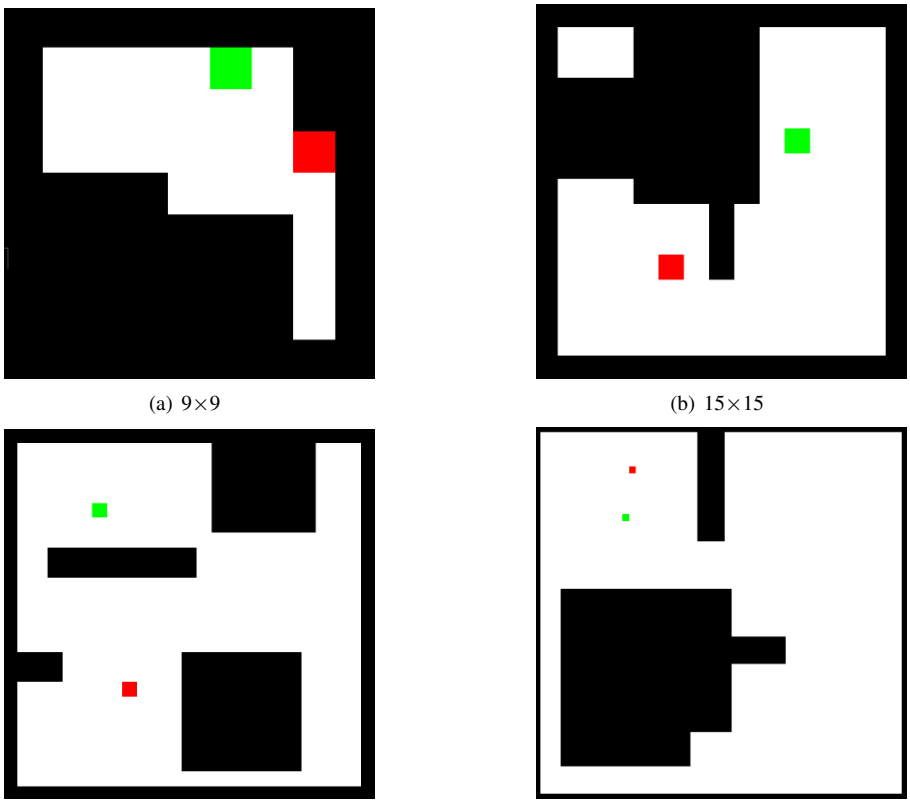


Fig. 5: The architecture of M-EVN, which includes convolution and max pooling operation.



(a) 9×9 (b) 15×15

(c) 35×35 (d) 55×55

Fig. 6: 2D Gridworld. (a) 9×9 gridworld, (b) 15×15 gridworld, (c) 35×35 gridworld, (d) 55×55 gridworld. From figure (a) to figure (d), the scale of Gridworld increases. The green box, the red box and the black area represent the agent, the target and obstacle area, respectively.

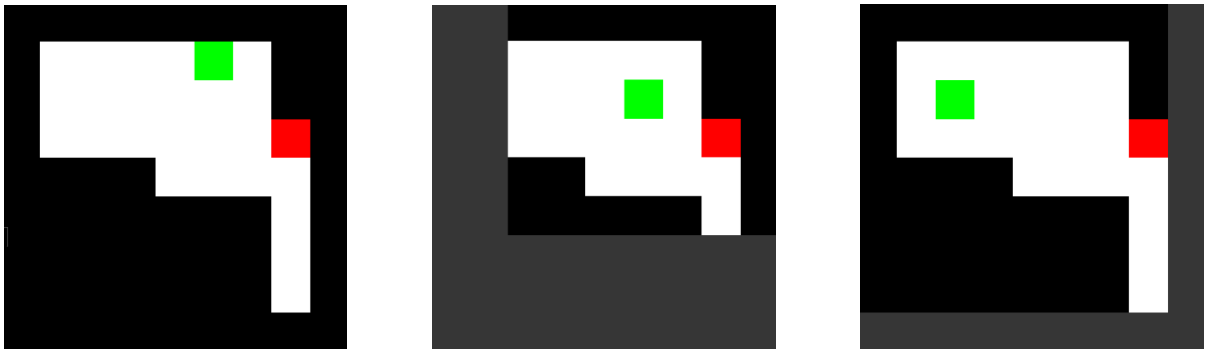
of each memory unit is 100. In addition, the learning rate is 2×10^{-3}, the discount ratio is 0.99, and the number of internal iterations is 10.

TABLE I: Hyperparameters in M-EVN

Learning rate	2×10^{-3}
Training times	2000
Discount ratio γ	0.99
DND capacity	500
DND size	200
Iteration number	10

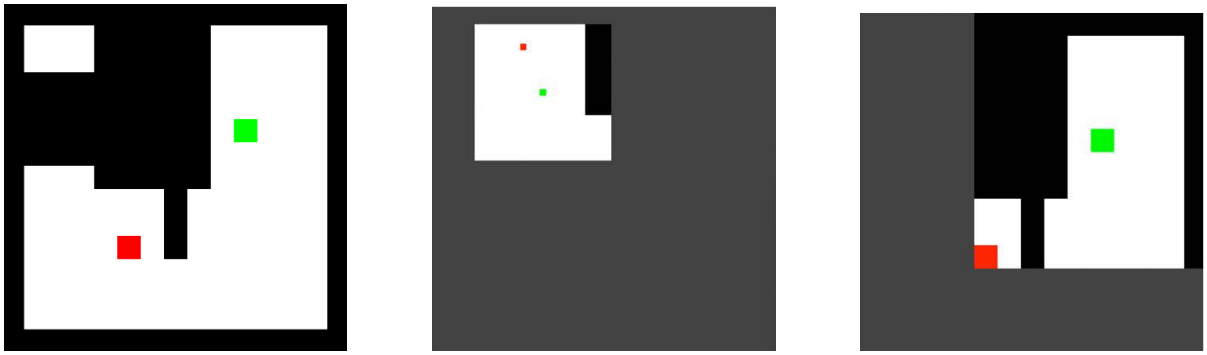
As shown in Fig. 6, the sizes of Gridwold game used in experiments are 9×9, 15×15, 35×35 and 55×55, respectively. For the navigation mazes, the navigation difficulty increases with the size of the maze. To facilitate the display of the mazes with different sizes, different mazes are scaled to the same size.

The green box represents the agent, the red box represents the target object, the black area along the edge of Gridworld is its boundary, and the black area inside Gridworld are obstacles. The agent's objective in Gridworld is to find the red box, and the agent has four available actions: turn left, turn right, move forward and move backward. If the agent reaches the correct target object, it succeeds; and if the agent collides with the obstacles, it fails.



(a) 9×9 (b) observation depth:3 (c) observation depth:5

Fig. 7: 9×9 maze. (a) fully observable maze, (b) 3×3 partially observable maze, (c) 5×5 partially observable maze.



(a) 15×15 (b) observation depth:3 (c) observation depth:5

Fig. 8: 15×15 maze. (a) fully observed environment, (b) 3×3 partially observed maze, (c) 5×5 partially observed maze.

TABLE II: Performance on 2D Gridworld with obstacles

Model	Depth	9×9	15×15	35×35	55×55
VIN	d3 (%)	3.16	0.02	0.83	0.12
	d5 (%)	**11.20**	0.78	0.92	0.12
M-EVN	d3 (%)	**6.50**	**4.74**	**0.85**	**1.00**
	d5 (%)	7.30	**2.56**	**0.97**	**1.80**

All the mazes used in the navigation experiments are partially observable, and the observation depth are 3 and 5, respectively. Taking 9×9 maze and 15×15 maze for example, as shown in Fig. 7, the white area represents the environment states which can be observed by the agent, while the black area represents the environment states which cannot be observed by the agent. Fig. 7(a) shows the fully observable environment of 9×9 maze, in which the agent can observe the states of the whole maze. Fig. 7(b) shows the maze environment with observation depth 3, and the black area is the environment states that the agent cannot observe. Fig. 7(c) shows the maze environment with observation depth 5. Compared with the maze with observation depth 3, the maze with observation depth 5 has smaller black area and larger white area, hence the agent can observe more environment states. Moreover, Fig. 8 shows the 15×15 maze, where the observable area decreases as the maze grows with certain observation depth.

B. Experimental Results and Analysis

The experimental results, as shown in Table II, show the navigation performance of VIN and M-EVN in 9×9, 15×15, 35×35 and 55×55 mazes with observation depth 3 and 5, respectively. In addition, the values in bold represent the higher accuracy in a maze with observation depth 3 and 5.

In Table II, except for the 9×9 maze with observation depth 5, the navigation performance of M-EVN agent is significantly better than that of VIN agent in the same maze with the same observation depth. Episodic memory stores past observations and absorbs new experiences into future actions. Specifically, M-EVN stores environmental features and state value function observed in the past into episodic memory. In the training period, M-EVN obtains the state with high similarity to the value function the current state from the episodic memory, calculates more reasonable value function estimation, and improves the navigation performance of the agent in partially observable environments.

The navigation performance of M-EVN is weaker than that of VIN in a 9×9 maze with observation depth of 5, due to the following reasons: the 9×9 maze is a simple maze, which is more like a fully observable environment with observation depth 5; VIN has better navigation performance in small and fully observable mazes. In fully observable

environments, especially in simple and small scale mazes, some useless information stored in episodic memory will affect the navigation policy of M-EVN.

Table II shows that M-EVN is more suitable for large scale and partially observable environments. M-EVN stores the environmental state features and the value functions of the past states in episodic memory. Meanwhile, value functions of the similar states are weighted and summed to obtain the value function of the current state. In essence, episodic memory stores the embeddings of past observations and assimilates new experiences into future behavior. Therefore, M-EVN has better navigation performance than VIN in partially observable environments.

V. CONCLUSION

In this paper, episodic memory is introduced into the value iterative network to enhance its navigation performance in partially observable maze games, and a memory-augmented episodic value network, called M-EVN, is proposed. M-EVN provides a memory-based value function estimation based on similar states, and constructs a Net-to-Net module embedded within to ensure the whole neural network differentiable during training period. We train M-EVN agent in partially observable maze games, and the experiments show M-EVN outperforms the original VIN in visual navigation.

REFERENCES

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[2] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[3] F. Zeng, C. Wang, and S. S. Ge, "Tutor-guided interior navigation with deep reinforcement learning," *IEEE Transactions on Cognitive and Developmental Systems*, 2020, doi:10.1109/TCDS.2020.3039859.

[4] D. Zhao, Y. Chen, and L. Lv, "Deep reinforcement learning with visual attention for vehicle classification," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 4, pp. 356–367, 2016.

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

[6] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[7] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu *et al.*, "Learning to navigate in complex environments," *International conference on learning representations*, 2017.

[8] A. Tamar, Y. Wu, G. Thomas, S. Levine, and P. Abbeel, "Value iteration networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 2154–2162.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[10] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[11] L. R. Squire, "Mechanisms of memory," *Science*, vol. 232, no. 4758, pp. 1612–1619, 1986.

[12] M. Fortunato, M. Tan, R. Faulkner, S. Hansen, A. P. Badia, G. Buttimore, C. Deck, J. Z. Leibo, and C. Blundell, "Generalization of reinforcement learners with working and episodic memory," *arXiv preprint arXiv:1910.13406*, 2019.

[13] J. Weston, S. Chopra, and A. Bordes, "Memory networks," *arXiv preprint arXiv:1410.3916*, 2014.

[14] S. Sukhbaatar, J. Weston, R. Fergus *et al.*, "End-to-end memory networks," in *Advances in neural information processing systems*, 2015, pp. 2440–2448.

[15] A. Pritzel, B. Uria, S. Srinivasan, A. P. Badia, O. Vinyals, D. Hassabis, D. Wierstra, and C. Blundell, "Neural episodic control," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 2827–2836.

[16] A. Graves, G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwińska, S. G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou *et al.*, "Hybrid computing using a neural network with dynamic external memory," *Nature*, vol. 538, no. 7626, p. 471, 2016.

[17] S. J. Gershman and N. D. Daw, "Reinforcement learning and episodic memory in humans and animals: an integrative framework," *Annual review of psychology*, vol. 68, pp. 101–128, 2017.

[18] C. Xiao, J. Mei, and M. Müller, "Memory-augmented monte carlo tree search," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[19] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[20] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[21] X. Li, M. Ye, Y. Liu, and C. Zhu, "Adaptive deep convolutional neural networks for scene-specific object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[22] J. Liu, R. Tan, G. Han, N. Sun, and S. Kwong, "Privacy-preserving in-home fall detection using visual shielding sensing and private information-embedding," *IEEE Transactions on Multimedia*, 2020.

[23] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7286–7291.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International conference on learning representations*, 2015.

[26] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[28] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[29] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[30] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856.

[31] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[33] T. A. Allen and N. J. Fortin, "The evolution of episodic memory," *Proceedings of the National Academy of Sciences*, vol. 110, no. Supplement 2, pp. 10 379–10 386, 2013.

[34] Adrien Turiot. gym-pathfinding. https://github.com/DidiBear/gym-pathfinding, 2018.