Modeling Attention-based Recall in an Egocentric Navigation Task

Team 9 (Jongmin Park, Sungyoung Kim)

1 Introduction

Memory is essential in reinforcement learning tasks for both humans and agents [1, 2, 3], where its direct implementation may uncover their behavioral insights. The project aims to investigate memory retrieval and its contextual details as learning progresses, with CNN-RNN-Attention-based memory recall architecture in a partially observable egocentric view and changing starting point navigation task. The code for model implementation can be found in the following link: `https://github.com/jongminpark88/Comp_neuro_final/tree/sungyoung`. The code for analysis and plots can be found in the following colab notebook: `https://colab.research.google.com/drive/1RnA6tRhTV3QbZtL0le_F7D3dcpkFZLlt?usp=sharing`.

2 Method

2.1 Environment

The environment is implemented using OpenAI Gym. At the start of each episode, the agent's initial position is randomly sampled, while the goal location remains fixed. This setup encourages the agent to actively utilize memory to achieve the goal from varied starting points.

We maintain a top-down view of the entire grid-world map (Figure 1). At each timestep, we crop the map around the agent's current position and perform ray casting on that cropped region to reconstruct a first-person, egocentric view. The resulting view is a 60×80 image representing what the agent perceives from its current location.

Each episode runs for up to 1,000 steps. The agent receives a penalty of -0.01 at every timestep, and upon reaching the fixed goal location, it receives a reward of $+100$ and the episode terminates.

2.2 Agent

The agent is a memory-augmented reinforcement learning agent. At each timestep, it processes its egocentric visual input and, when appropriate, uses recalled memories to decide on an action. The action space consists of 'turn left,' 'turn right,' and 'move forward' based on the agent's current heading. Note that the agent's observation is the egocentric view obtained via ray casting and absolute map coordinates are not provided directly.

2.3 Model Architecture

The overall model consists of two main components: the policy network and the memory bank (Figure 2). At each timestep, the agent's experience is added to the memory bank, and the policy network uses both the current observation and recalled memory to select the agent's next action.

2.3.1 Memory Bank

The memory bank serves as a repository of past experiences, organized into slots. Each slot stores episode number, agent's grid position, RNN hidden state and action taken at that timestep, as well as an embedding for past observations.

In total, the memory bank holds slots for up to 5 complete episodes (5,000 slots). When capacity is exceeded, memories are deleted according to a priority score. Each time a memory is recalled, its priority increases; otherwise, priority decays over time.

2.3.2 Policy Network

The policy network is comprised of CNN and RNN layers (Figure 2 Policy Network). The egocentric view is passed through a sequence of CNN layers to extract spatial features, producing current observation embedding. This embedding is then combined with the previous RNN hidden state and passed through an RNN layer to yield the current hidden state. The resulting hidden state is merged with recalled memory via a learnable gating mechanism (implemented as a small MLP). By dynamically weighting current sensory information against past memories, the gate allows the network to select whichever source of information is most useful for deciding the action at each timestep. Finally, the combined vec-

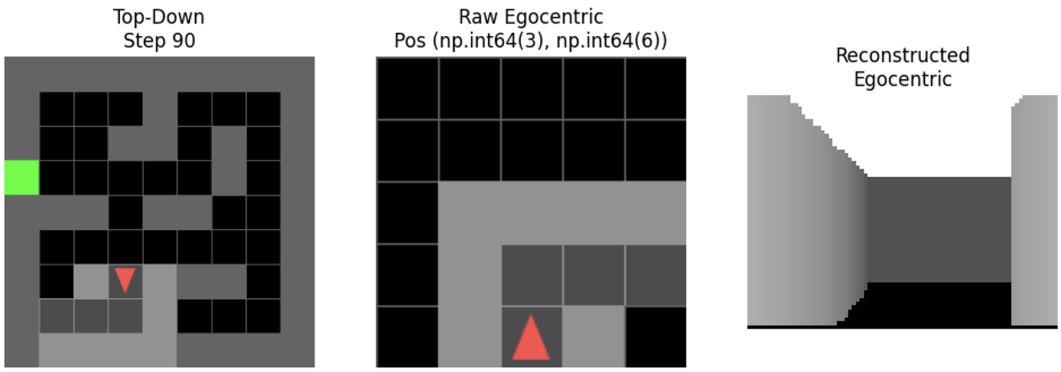

Figure 1: Environment overview at timestep 90
(a) full top-down map (b) agent-centric crop (c) reconstructed egocentric 60×80 view

tor goes through a fully connected layer to produce action probabilities.

2.4 Recall

At each timestep, the recalled memory vector h_{memory} is computed via an attention mechanism over all stored memory slots (Figure 2 Memory Bank). First, we compute a similarity score between the current observation embedding and each memory slot's stored observation embedding. Based on these similarity scores, we select the top five most similar memory slots. Next, we apply a softmax over their similarity values to obtain normalized attention weights. Using these weights, we compute a weighted sum of the selected slots' hidden states to form the recalled memory vector. Finally, this recalled memory m_t is merged with the current hidden state h_t via a MLP-learned gating parameter α_t to produce a combined hidden state $h_t^{\text{combined}} = \alpha_t\, m_t + (1-\alpha_t)\, h_t$.

2.5 Training Algorithm

We experimented with two policy-gradient methods.

2.5.1 REINFORCE

For each episode we collect the full trajectory $\{(s_t, a_t, r_t)\}_{t=0}^{T}$ where the return G_t is calculated as $G_t = \sum_{k=t}^{T} \gamma^{k-t}\, r_k$. We then update the policy parameters θ according to $\Delta\theta \propto \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\,\big(G_t - b\big)$, where b is the average return over recent episodes. A known drawback of REINFORCE is its high variance in gradient estimates, which can lead to unstable learning.

2.5.2 A2C

In A2C, we train both an actor (the policy network π_θ) and a critic (a value-function approximator $V_\phi(s)$). At each timestep, we compute the one-step temporal-difference (TD) error $\delta_t = r_t + \gamma\, V_\phi(s_{t+1}) - V_\phi(s_t)$ based on which we perform actor update $\Delta\theta \propto \nabla_\theta \log \pi_\theta(a_t \mid s_t)\, \delta_t$ and critic update $\Delta\phi \propto \delta_t\, \nabla_\phi V_\phi(s_t)$.
The critic's role is to provide a lower-variance estimate δ_t of how much better or worse the received reward was compared to expectation. Note that both REINFORCE and A2C are standard training algorithms and we outlined them for the purpose of clarity.

3 Results

We report cumulative success rate for REINFORCE and A2C algorithms in the 9 x 9 and 13 x 13 maze environments (Figure 3), where A2C showed a slower learning than REINFORCE in this particular setting. Based on the learned agent, we aim to investigate the following research questions on memory retrieval and context:

- **RQ1: Memory Retrieval** How does the degree of memory retrieval change as learning progresses?
- **RQ2: Memory Context** What combination of memory and context impacts learning?

3.1 RQ1: Memory Retrieval

We first plotted gate alpha across timesteps. In the initial episode, gate alpha remained nearly constant as opposed to when episode progressed, where gate alpha fluctuated over timesteps (Figure 4), indicating memory retrieval weight is being adjusted across time in an episode. Gate alpha was additionally averaged across episodes and plotted against timesteps in a similar manner (Figure 5), which showed its decrease within 150 - 400 steps followed by an increase

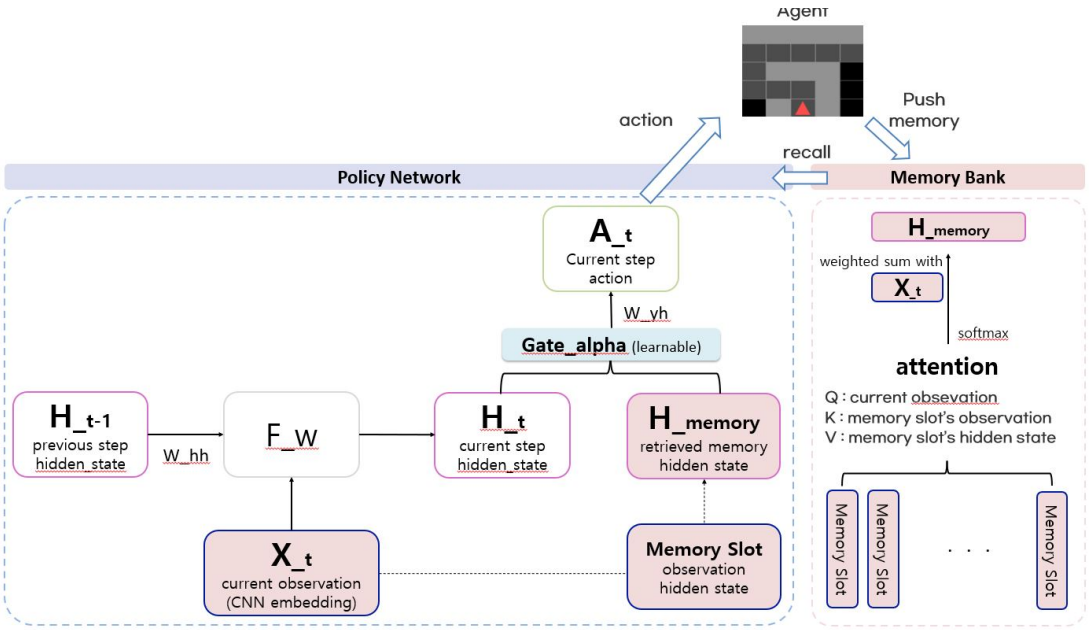

Figure 2: Full Architecture of Policy Network, Memory Bank and their interaction with agent

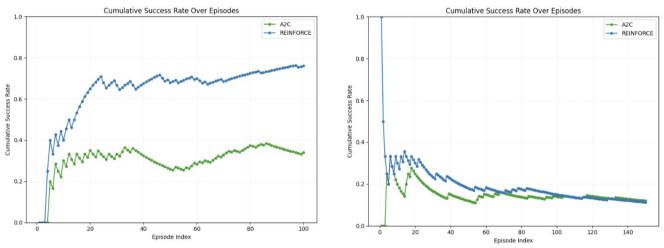

Figure 3: cumulative success rate
(a) 9 x 9 maze (up to 100 episodes) (b) 13 x 13 maze (up to 150 episodes)

within 400 - 600 steps and stabilization from 600 steps onwards. This is another sign of memory retrieval weight adjustment where observation may be more reliable than memory early in the learning, but the importance of memory increases as learning progresses and enough memory is filled.

We next investigated gate alpha across episodes. Similar to the across-timesteps analysis, it stayed nearly zero at the start, which eventually stabilized, but decreased for the last 200 episodes. When gate alpha was nearly zero, cumulative success rate increased, and when gate alpha stabilized or decreased, so did success rate, suggesting their in-phase pattern as well as the importance of gate alpha in the task success. Note that the decrease in success rate is likely to be the result of overfitting.

We further investigated attention weight parameter over episodes. Absolute attention weight (Figure 7 (a)) was nearly zero up to 200 episodes where it peaked and stabilized to a non-zero value. Its standard deviation (Figure 7 (b)) showed a similar pat-

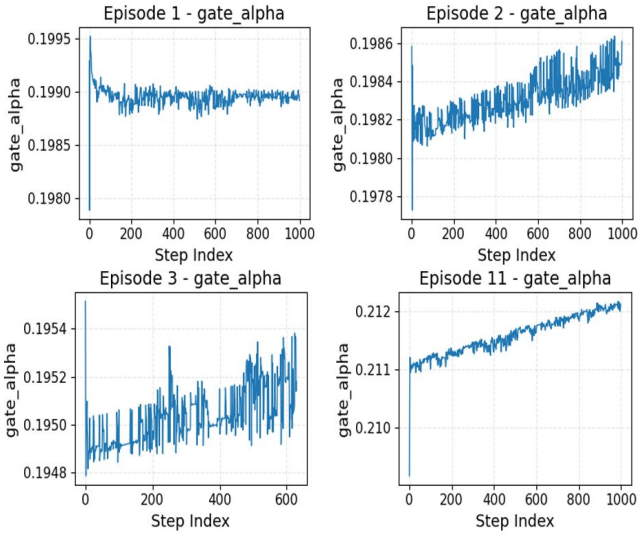

Figure 4: gate alpha change across timesteps
(a) episode 1 (b) episode 2 (c) episode 3 (d) episode 11

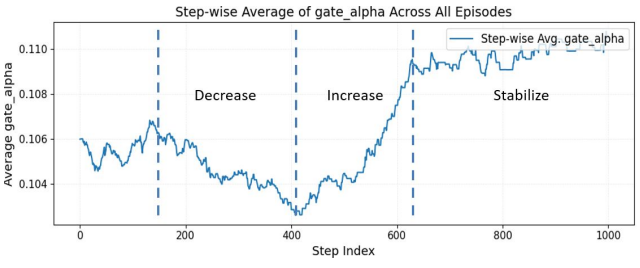

Figure 5: Gate alpha across timesteps (averaged across episodes)

tern, both of which indicate that attention began to function as learning paced.

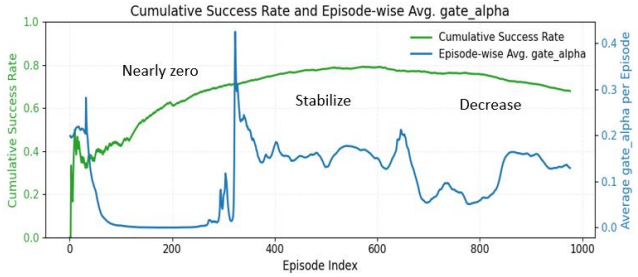

Figure 6: Cumulative success rate and gate alpha across episodes (averaged across timesteps)

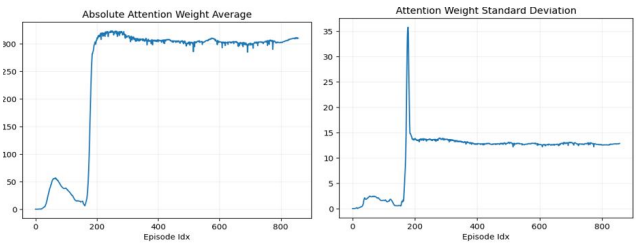

Figure 7: (a) Absolute attention weight (b) Attention weight standard deviation, both averaged over timesteps, plotted against episodes

We finally investigated distance between observed location and retrieved memory location, where it abruptly increased at the start as well as showing a marginal increase towards later episodes, indirectly showing how policy network gradually referred to a memory at a further away location starting from nearby locations. The linear regression results of the moving averaged distance of the last half of episodes (from 500 episodes onwards) showed a slope of 0.0016 and the correlation with episode index of 0.77, with p-value $<$ e$-$90, where the marginal slope indicates further analysis will be favored to verify the provided insights.

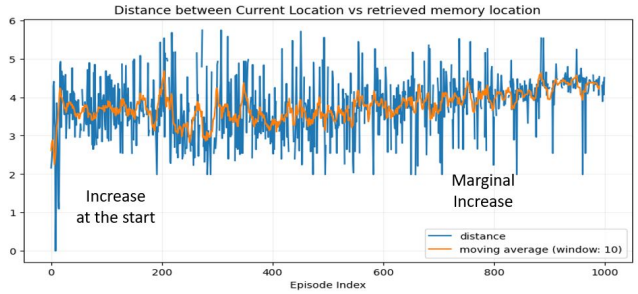

Figure 8: Distance between observed location and memory-retrieved location

In summary, gate alpha across timesteps and episodes, with attention weight parameter and the observed-retrieved location distance developing across episodes suggest (1) memory retrieval taking place as agent learning progresses and (2) while observation dominates in the early phase of learning, the importance of memory retrieval increases and stabilizes later, in phase with success rate or attention weight, appearing to enable the agent to refer to memories that are beyond its vicinity.

3.2 RQ2: Memory Context

The top 10 and bottom 10 locations of memory that were most frequently retrieved are denoted in Figure 9. Note that the top 10 memory-retrieved locations tend to be in the middle of the corridor, further away from goals (outside the white box) while the bottom 10 memory retrieved locations tend to be in the corners, closer to the goal (inside the white box).
We further looked into the locations where the top 5 and bottom 5 memory retrieval took place. In Figure 10 (a), the purple box indicates location that has been retrieved as memory most frequently and the color of the heatmap indicates at which location this memory has been retrieved (e.g. agent at white position retrieved memory collected at purple). It can be seen that the purple memory was frequently retrieved when the agent was at corners further away from the goal. Although Figure 10 (b) suggests a less noticeable pattern for two bottom 10 memory-retrieved locations, it appears that the agent repetitively recalls wide-range of memories across the maze.

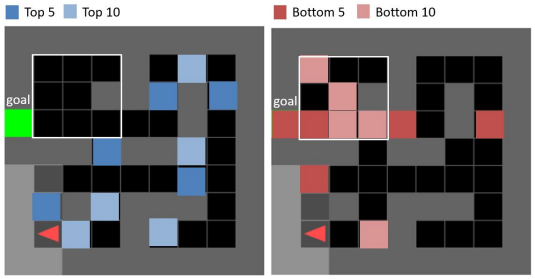

Figure 9: Top (resp. Bottom) 10 locations most (resp. least) frequently retrieved

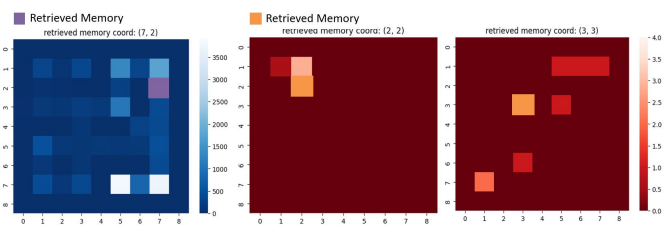

Figure 10: Heatmap showing the number of each location's retrieval of specified memory

4 Discussion

This project studied agent's behavior in relation to its memory retrieval and context, based on (1) the egocentric view and changing starting point navigation task design and (2) direct memory retrieval mechanism implemented with CNN + RNN embedding + Attention. While we suggested evidence for memory retrieval adjustment/stabilization where memories further away from goals were retrieved at locations not necessarily close to them across the maze, "do humans truly behave as RL agents in an egocentric view navigation task?" or "why do agents recall memories in such a way" remain as a room for further investigation, where we concretize its prospective methods and related work as follows:

- Human-analogous memory retrieval: intentionally simplified or disrupted memory representation ([5])
- Discovering human memory retrieval mechanism: supervised human cloning to reveal human memory retrieval mechanism which goes beyond agent simulation ([6])

References

[1] Jensen, K.T., Hennequin, G., Mattar, M.G. (2024). A recurrent network model of planning explains hippocampal replay and human behavior *Nature Neurosciences*

[2] Frankland, P.W., Josselyn, S.A., Köhler, S. (2019) The neurobiological foundation of memory retrieval *Nature Neurosciences*

[3] Michael T. Todd, Yael Niv, Jonathan D. Cohen (2008) Learning to use Working Memory in Partially Observable Environments through Dopaminergic Reinforcement *NeurIPS*

[4] Sutton, R. S., Barto, A. G. (2018) Reinforcement learning: An introduction (2nd ed.) *The MIT Press*

[5] Ho, M.K., Abel, D., Correa, C.G (2022) People construct simplified mental representations to plan *Nature*

[6] Eckstein, M.K., Summerfield, C, Daw, N, Miller, K.J. (2024) Hybrid Neural-Cognitive Models Reveal How Memory Shapes Human Reward Learning *PsyArXiv*

*GPT 5.1 for the write-up and initial coding