Rapid Learning without Catastrophic Forgetting in the Morris Water Maze

Raymond L Wang Jaedong Hwang Akhilan Boopathy Ila Fiete
Massachusetts Institute of Technology
`{rlwang,jdhwang,akhilan,fiete}@mit.edu`

Abstract

Machine learning models typically struggle to swiftly adapt to novel tasks while maintaining proficiency on previously trained tasks. This contrasts starkly with animals, which demonstrate these capabilities easily. The differences between ML models and animals must stem from particular neural architectures and representations for memory and memory-policy interactions. We propose a new task that requires rapid and continual learning, the sequential Morris Water Maze (sWM). Drawing inspiration from biology, we show that 1) a content-addressable heteroassociative memory based on the entorhinal-hippocampal circuit with grid cells that retain knowledge across diverse environments, and 2) a spatially invariant convolutional network architecture for rapid adaptation across unfamiliar environments together perform rapid learning, good generalization, and continual learning without forgetting. Our model simultaneously outperforms ANN baselines from both the continual and few-shot learning contexts. It retains knowledge of past environments while rapidly acquiring the skills to navigate new ones, thereby addressing the seemingly opposing challenges of quick knowledge transfer and sustaining proficiency in previously learned tasks.

1 Introduction

Animals can *rapidly* learn new tasks that are conceptually similar to previously encountered tasks, but have different inputs and surface-level details; simultaneously, *they retain the ability to solve the previous tasks*. Neural modeling of this process of rapid conceptual knowledge transfer with retention of past learning has been limited. In some ways, rapid learning and learning retention seem to be in opposition: the former requires fast adaptation of parameters while the latter requires stable parameters. In machine learning, models tend to focus on either solving rapid learning and transfer, or on continual learning without forgetting; models tend not to do well at both.

Here, we build a biologically motivated neural model to solve a sequential version of the classic Morris Water Maze task (6; 12), in which a rodent must find and then navigate to a submerged platform in a pool of cloudy water across multiple trials starting from different positions. We term our variant of this task the Sequential Morris Water Maze (sWM) task. This task necessitates sequential learning across multiple unique environments, each characterized by a different platform location. Within a single environment, the task demands two generalizations from the agent. First, it must generalize its learning from a variety of starting locations. Second, it must rapidly adapt to changes in the goal locations. In our sequential version of the task, an additional layer of complexity is introduced. Here, the agent is required to learn new environments while preserving the knowledge of the previous ones. This requirement tests the agent's ability to avoid catastrophic forgetting, a significant challenge in machine learning. Thus, the sWM task not only involves the aforementioned *intra-environment* generalization and adaptation but also *inter-environment* learning and memory retention.

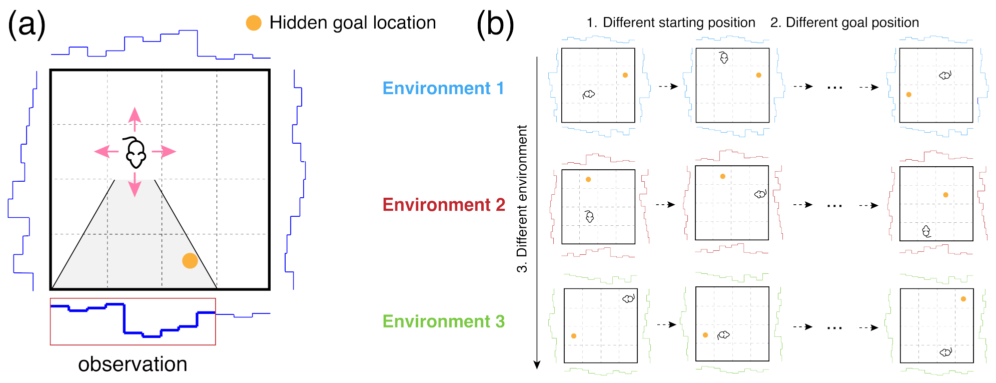

Figure 1: Schematic of the sequential Morris Water Maze task. **(a) The water maze environment.** The rodent icon represents the agent, arrows indicate the rodent's allowed actions, gold circles indicate the hidden platforms, and curves parallel to the walls of each environment denote patterns along the walls. The agent observes a portion of the wall pattern. **(b) our training setup.** Agents undergo three tests of generalization. 1. Training sequentially over multiple trajectories in a single environment with a fixed goal location. 2. Training across multiple trajectories and goal locations within one environment. 3. Training in various environments, each with multiple trajectories and goal locations. The agent is evaluated on rapidly learning to navigate in new goal locations and new environments and remembering navigation strategies in previously seen environments.

Conventional unstructured neural networks suffer from catastrophic forgetting: a phenomenon where networks trained on a sequence of tasks fail to perform well on previously trained tasks (5). Unstructured neural networks generally also lack an intrinsic ability to generalize rapidly to unseen tasks. Networks that perform rapid task transfer are typically extensively trained on a large number of related tasks (e.g. using multi-task learning techniques (1) or meta-learning (11)).

We propose that certain inductive biases, like those present in the brain, allow networks to avoid these shortcomings and achieve performance on rapid and continual learning that is comparable to animals. It is known that animals use specialized computations in the hippocampus and entorhinal cortex to enable efficient spatial navigation and learning (7; 2). We use a structured neocortical-entorhinal-hippocampal circuit, the Memory Scaffold with Heteroassociation (MESH) architecture (9), to enable such generalization in the Water Maze *after training only on a single environment*. Our model proceeds as follows: first, MESH maps observation signals to a *grid cell* pattern, a type of spatial representation found in the entorhinal cortex. The grid code is then inputted into a randomly initialized, fixed Convolutional Neural Network (CNN), yielding a spatially invariant output feature vector. Lastly, this feature is processed by an attention module to determine the agent's action.

Our approach integrates a high-capacity content-addressable memory system with a spatially-invariant network specifically designed to facilitate zero-shot policy learning in new environments. Conceptually, this combination is beneficial as it allows the system to store and retrieve relevant information efficiently, while also adapting rapidly to new environments without requiring additional training. This functionality reflects the learning behavior of biological entities, contributing to the agent's capacity for both knowledge retention and rapid, flexible learning. We would like to emphasize that we are the first work that employs MESH in continual learning tasks.

2 Morris Water Maze Task

We've developed the Sequential Morris Water Maze (sWM) task, a modification of the traditional Morris Water Maze. While the classic task measures a rodent's capacity to recall past environments and adapt to new ones using a circular tub and distal cues, our variant employs a square tub with unique wall markings as cues. These cues are processed as vector inputs, aiding the agent in navigation to find a hidden platform. The agent can move in four directions: north, south, east, and west.

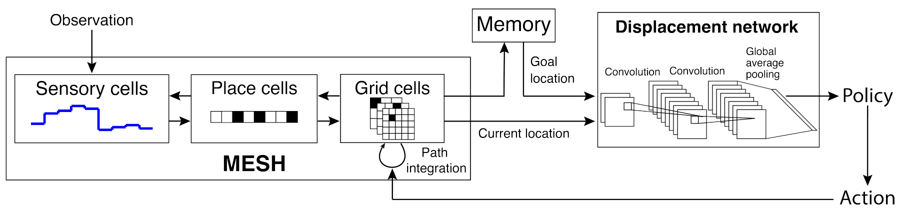

Figure 2: Schematic of our model: MESH for Spatial Navigation (MSN) The agent observes a portion of wall. Observations, along with velocity inputs, are fed into a MESH network that produces grid cell activations representing the agent's location. An external memory module stores the grid code of the goal location. Grid codes of the current location and goal location are fed to the displacement network, a spatially-invariant convolutional neural network to produce a representation of the relative goal position. This is fed to a policy that produces actions.

After mastering one environment, the agent undergoes sequential training. Here, it encounters both known and new environments from varied starting positions. This increases the challenge, compelling the agent to adjust its tactics based on its starting point and the platform's location.

Our sWM task evaluates the agent's cognitive skills, emphasizing memory retention and adaptability. Just as rodents efficiently display these traits, our goal with the artificial agent is to mimic this proficiency in navigating intricate tasks.

3 MESH for Spatial Navigation (MSN)

3.1 Motivation and Overview

Artificial neural networks, despite their advancements, suffer from 'catastrophic forgetting' during continual learning. This flaw makes them forget older tasks when trained on new ones. In contrast, biological entities like rodents and humans resist such forgetting, showcasing the complexity of biological learning systems. Grid and place cells in the hippocampus, are useful in preventing catastrophic forgetting, especially for spatial memory. They're thought to form cognitive maps, aiding navigation and spatial recall.

Drawing inspiration from this, we introduce a continual learning approach based on MESH (9), termed MESH for Spatial Navigation (MSN). We describe some details of MESH in Appendix A Initially, the MESH translates observations to grid cell patterns or 'grid codes', emulating grid cells in the brain vital for spatial understanding. This code feeds into a fixed Convolutional Neural Network (CNN), chosen for its spatial invariance and outputs a feature vector. Lastly, an attention module processes the feature vector, selecting actions based on previously seen features.

3.2 Associating Grid Code Displacements with Movements

We develop a model of how rodents rapidly learn to navigate in new environments. Using a randomly initialized fixed convolutional neural network (CNN), our model maps the rodent's current and goal locations (encoded in a grid code, provided by MESH) to a spatially-invariant representation of the *displacement* of the goal relative to its current position. We use an attention mechanism with the keys being the spatially-invariant representation of the grid code and the values being the appropriate actions. During the training phase, these key-value pairs are associated and stored within the mechanism. During the testing phase, the agent's current state generates a spatially invariant representation of displacement that is used as the query. This query is then processed through a dot product operation with the existing keys in the dictionary. The action associated with the key most similar to our query is identified and used. This process allows for efficient action selection based on the spatially invariant displacement of the agent. Our architecture's spatial invariance allows the agent to rapidly learn to navigate in unseen environments *by only learning associations between new observations and the grid code* (it does not need to learn new associations to actions) as we will discuss in the next section. Figure 2 illustrates our model architecture.

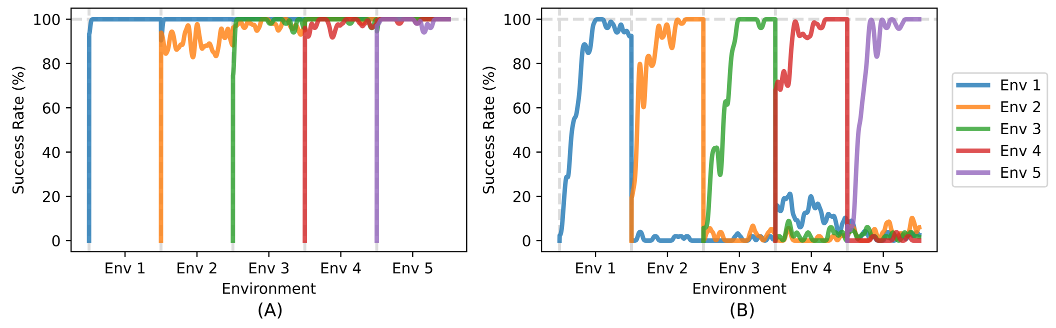

Figure 3: Our model avoids catastrophic forgetting. The average success rate of each environment while training on the following environments of our final model (a) and ours without MESH (b). The full model rapidly outperforms the one without MESH. In both plots, we use a moving average of 25 points and Gaussian smoothing with $\sigma = 10$.

3.3 Agent Training and Zero-Shot Policy Learning in Novel Environments

The agent is now equipped with two key features: resistance to catastrophic forgetting and adaptability to new environments, both of which enhance the learning process. In its initial training, the agent explores a novel environment, gathering data. This data is linked with a grid code through the MESH framework. After identifying the goal's position, its grid code is saved, representing the goal on its cognitive map. Using our spatially invariant CNN, a vector for displacement is computed. An attention mechanism associates this 'displacement' to action, allowing the agent to select the correct action based on previously learned distances.

In new environments, once the goal is found, the agent's policy requires no further training. It calculates the goal's distance using the spatially invariant CNN, then uses its attention mechanism to navigate. This approach facilitates zero-shot policy learning in unfamiliar terrains, highlighting our framework's adaptability and efficiency.

4 Experiments

In Figure 3, our approach (a) exhibits rapid learning in the first environment compared to the baseline neural network trained in a fine tuning framework shown in (b), where the observations are fed directly into a neural network and supervised by the correct action. Furthermore, our method successfully acquires a general, transferable navigation policy from this initial environment, allowing rapid navigation in subsequent environments *without any policy training*. This contributes to the prevention of catastrophic forgetting, as past environments can be recalled after recognizing the current environment through a few trajectories. In sharp contrast, the baseline experiments demonstrate an almost immediate onset of catastrophic forgetting upon exposure to a new environment. This phenomenon is marked by a rapid performance decline following the training of a few new trajectories, despite the initial successful knowledge transfer and adequate performance in the new setting. We further test our method against other continual learning baselines in Appendix B and individual components of our model in Appendix C.

5 Discussion

We present a new neural model based on the MESH architecture. It excels in rapid learning across various environments and transfers knowledge effectively to new scenarios. Unlike traditional machine learning methods which struggle with rapid learning and avoiding 'catastrophic forgetting', our model adeptly addresses these issues, highlighting the advantages of integrating inductive biases.

Our results emphasize the potential of brain-inspired structured neural models in AI and neuroscience. They indicate the capacity of such models to redefine AI capabilities. Future work should focus on optimizing our model, expanding its applicability, and assessing its performance in diverse, dynamic settings.

References

[1] Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

[2] Torkel Hafting, Marianne Fyhn, Sturla Molden, May-Britt Moser, and Edvard I Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806, 2005.

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[4] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13):3521–3526, 2017.

[5] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. 1989.

[6] Richard GM Morris. Spatial localization does not require the presence of local cues. *Learning and motivation*, 12(2):239–260, 1981.

[7] John O'Keefe and Jonathan Dostrovsky. The hippocampus as a spatial map: preliminary evidence from unit activity in the freely-moving rat. *Brain research*, 1971.

[8] L Personnaz, I Guyon, and G Dreyfus. Information storage and retrieval in spin-glass like neural networks. *Journal de Physique Lettres*, 46(8):359–365, 1985.

[9] Sugandha Sharma, Sarthak Chandra, and Ila Fiete. Content addressable memory without catastrophic forgetting by heteroassociation with a fixed scaffold. In *ICML*, 2022.

[10] Jonathan Tapson and André van Schaik. Learning the pseudoinverse solution to network weights. *Neural Networks*, 45:94–100, 2013.

[11] Sebastian Thrun and Lorien Pratt. Learning to learn: Introduction and overview. *Learning to learn*, pages 3–17, 1998.

[12] Charles V Vorhees and Michael T Williams. Morris water maze: procedures for assessing spatial and related forms of learning and memory. *Nature protocols*, 1(2):848–858, 2006.

[13] Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Deep class-incremental learning: A survey. *arXiv preprint arXiv:2302.03648*, 2023.

Appendix

A MESH

MESH (9) is a content-addressable memory (CAM) model based on the architecture of the neocortical-entorhinal-hippocampal memory circuit in the brain. Content-addressable memory models are networks that can store vectors (patterns to be memorized) as fixed points of their dynamics, and thereby recall/reconstruct them from noisy cues. Specifically, given a corrupted version of a previously encountered pattern, CAM models aim to reconstruct the original un-corrupted ground truth pattern. CAM models often suffer from a memory cliff problem: when the number of stored patterns crosses a certain threshold, the model not only fails to learn any new patterns, but also abruptly fails to recall all previously stored patterns. This is a form of *catastrophic forgetting*.

MESH addresses the memory cliff problem by constructing a fixed scaffold of pre-defined content-independent fixed points, which are then used to store the content-laden patterns through hetero-associative learning, thus mimicking the neocortical-entorhinal-hippocampal circuit to store patterns. The MESH architecture consists of three layers; features, hidden states, and labels, which biologically correspond to sensory input, place cell layer, and grid cell layer, respectively. We use grid code as labels instead of the k-hot labels in MESH. The place cell layer $\mathbf{p} \in \{-1, +1\}^{N_P}$ represents an N_P dimensional binary vector, the grid cell layer $\mathbf{g} \in \{0, 1\}^{\sum \lambda_i}$ is defined as the concatenation of λ_i dimensional one-hot vectors each of which represents a grid module in the brain, and the sensory layer is N_s dimensional.

Before starting experiments, the memory scaffold (grid and place cells sates, as well as the projections between the grid and place cell layers) is *pre-defined*. The projection matrix from the grid cell layer to the place cell layer, $\mathbf{W}_{PG}$ is randomly generated so that it maintains an injective projection. On the other hand, the weight matrix from the place cell layer to the grid cell layer is trained by Hebbian learning such that it associates each active place cell (defining a place code) to the concurrently active grid cells (defining a corresponding grid code):

$$\mathbf{W}_{GP} = \frac{1}{|\mathbf{N}|} \sum_{\mu=1}^{\mu=N} \mathbf{g} \cdot (\mathsf{sign}(\mathbf{W}_{PG} \cdot \mathbf{g}))^T, \tag{1}$$

where N is the number of training patterns.

When the agent explores the environment, the weights between sensory inputs and the place cells ($\mathbf{W}_{SP}$ and $\mathbf{W}_{PS}$) are learned by a pseudoinverse learning rule (8) in an online manner (10), yielding the following final weights:

$$\mathbf{W}_{SP} = \mathbf{S} \cdot \mathbf{P}^{\dagger}, \tag{2}$$

$$\mathbf{W}_{PS} = \mathbf{P} \cdot \mathbf{S}^{\dagger}, \tag{3}$$

where $\mathbf{S}$ and $\mathbf{P}$ are $N_s \times N$ and $Np \times N$ dimensional matrices of sensory patterns and place patterns respectively, and † indicates the pesudoinverse.

In summary, given the sensory input $\mathbf{s}_t$ at time t, the corresponding place cell and grid cell activations are computed through the model dynamics as follows:

$$\mathbf{p}_t = \mathsf{sign}(\mathbf{W}_{PS} \cdot \mathbf{s}_t), \tag{4}$$

$$\mathbf{g}_t = \mathsf{CAN}(\mathbf{W}_{GP} \cdot \mathbf{p}_t). \tag{5}$$

where $\mathsf{CAN}(\cdot)$ represents the continuous attractor recurrence in the grid layer that is implemented using a module-wise winner-take-all dynamics. This ensures that the equilibrium grid state is always a valid grid code i.e., a concatenation of one-hot vectors corresponding to each grid module.

The grid cell layer receives velocity signals (action input $\mathbf{a}_t$) for path integration, where the activated index for each grid cell module is shifted according to the action direction to infer the next grid state. Once we obtain the next grid code $\mathbf{g}_{t+1}$, its corresponding place code $\mathbf{p}_{t+1}$ is associated with the sensory input ($\mathbf{s}_{t+1}$).

In our implementation, we extend the grid cell modules to 2D space (with λ_i^2 dimensions for each one-hot grid cell module) and adapt the path integration described above to suit the proposed 2D sequential Morris Water Maze environments.

B Experimental Details

We optimize parameters using Adam (3) with a learning rate of 0.001 for 800 episodes for each environment. The maximum number of steps in each episode is set to 100 and the starting configuration (head direction and coordinates) are different. The environment is a 30×30 grid with unique, noise-added step function markings on the walls. The agent has a field of view (FOV) of 120 degrees (see Figure 1a). We use a public continual learning implementation (13) for EWC (4) and implement our own version of replay buffer and fine-tuning. For fine-tuning, we sequentially train on each environment. In our replay buffer implementation, we allocated a fixed buffer size (100 in our case) during the training of the neural network within a single environment. Throughout this training phase, we stochastically selected data points for inclusion in our replay buffer. Upon completion of training in one environment, we initiated a fine-tuning process on our replay buffer by sampling from it, followed by an evaluation in all previously trained environments. This procedure was replicated across all five environments. We also examined the sensitivity to replay buffer size.

To address this shortcoming of the baseline, we employed strategies additional strategies in continual learning such as the use of a replay buffer and Elastic Weight Consolidation (EWC) on the baseline neural network. Despite these efforts, both the replay buffer strategy and EWC demonstrated signs of catastrophic forgetting. Figure 4 displays the average success rate of all previously trained environments after training on the environment indicated on the x-axis. Our method consistently outperforms the continual learning baselines, whereas other methods exhibit degraded performance as more environments are introduced for training. Table 1 shows our method compared to baselines on all five environments after all training is complete.

The underwhelming performance of EWC in our tasks appears to stem from the similarity of inputs across different environments. Despite these similarities, the goal positions differ between environments. Consequently, similar observations could map to two distinct actions. EWC aims to maintain the weights of the network to find an overlap between all tasks. However, due to this subtle complexity in our task design, EWC fails to perform optimally.

C Ablation Study

The effectiveness of each individual component in our proposed method is analyzed and summarized in Table 2 in Appendix. Overall, the attention module plays a crucial role in achieving high

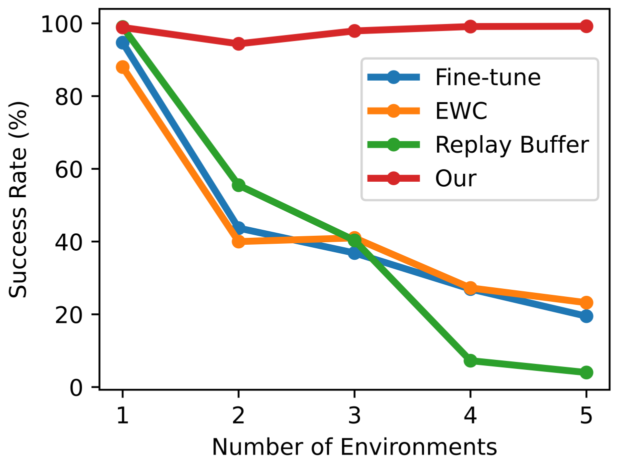

Figure 4: The average success rate along incremental stages. Our method clearly outperforms the continual learning baselines maintaining almost performance while other methods are degraded as training goes on.

Table 1: The average success rate (%) of each environment after training all environments. Our model maintains high success rates across the entire environment while other methods have bad performance except for the last environment (Env 5) due to catastrophic forgetting.

Training Scheme	Average	Env 1	Env 2	Env 3	Env 4	Env 5
Fine-Tune	19.5	2.2	3.6	2.4	0.5	99.9
EWC (4)	23.2	0.0	0.0	16.0	0.0	**100.0**
Replay Buffer	4.0	0.0	9.0	0.0	3.0	8.0
Ours	**99.2**	**99.2**	**99.5**	**99.0**	**99.5**	98.8

performance. In fact, when used alone, the attention module achieves a perfect success rate. This is because when the goal location remains fixed, there is no need to rely on the spatial invariance provided by CNN (policy). Instead, the grid code can be directly associated with the attention module. This approach must learn associations between *every* observation and the corresponding ground-truth actions, which is memory inefficient and non-transferable to new environments. Furthermore, this approach becomes vulnerable when there are changes in the goal locations within the same environment since associations between observations and actions must be re-learned. On the other hand, when the attention module is combined with MESH without CNN, the performance is significantly lower. This is likely because MESH lacks spatial invariance, leading to the learning of conflicting associations between the grid code and actions. As for the CNN without the attention module, it corresponds to the "Fine-Tuning" model presented in Table 1. The use of MESH allows the CNN to use different input encoding methods, enhancing its versatility.

In summary, the superior accuracy demonstrated by the encoding network with attention, or by the attention mechanism in isolation, can primarily be attributed to its perfect memorization capabilities. This becomes apparent when the attention mechanism undergoes training as it is just storing key-value pairs. However, in the absence of such training, the model's performance in future environments significantly declines, often nearing zero. This reveals a lack of forward transfer or generalization capabilities in the model.

To verify the effectiveness of spatial invariance from CNN, we train one environment with the fixed goal location and evaluated it with the changed goal location. Table 3 shows that all three modules should be combined together to solve the new location with further training. Furthermore, we also test that using fully-connected layers (FC) instead of CNN cannot solve the problem, which emphasizes the need to spatial invariance to find unseen goal locations.

Table 2: Combined components of MSN allows for zero-shot transfer to new environments. The last row is our final model. We measure the average success rate (%) across all environments after training the last environment. The exception is the case of MSN: MESH, the encoding network, and attention mechanism, where the system is trained in a single environment and subsequently tested across five different environments.

MESH	Policy	Attention	Training Environment ID	Success Rate
		✓	1, 2, 3, 4, 5	100
	✓		1, 2, 3, 4, 5	19.5
	✓	✓	1, 2, 3, 4, 5	99.7
✓	✓		1, 2, 3, 4, 5	0.9
✓		✓	1, 2, 3, 4, 5	5.4
✓	✓	✓	1	99.2

Table 3: Our model allows for adaptation to new goal locations not included during training. The last row is our final model, MSN. We measure the average success rate (%) of a new goal location after training the different goal in one environment. The exception is the case of MESH, the encoding network, and attention mechanism, where the system is trained in a single environment and subsequently tested across five different environments.

MESH	Policy	Attention	Success Rate
		✓	1.6
	✓		2
	✓	✓	1.8
✓	✓		1
✓		✓	5.6
✓	✓(FC)	✓	0.1
✓	✓	✓	99.5

Our framework, which includes MESH, the encoding network, and the attention mechanism, is trained exclusively on one environment and subsequently evaluated on four *unseen* environments and one seen environment. Conversely, all other ablated models are trained and then evaluated in all five environments. We adopted this strategy due to the observation that, without any training in future environments, each of our ablation study networks merely exhibited random movement, demonstrating no ability to generalize.

D Dynamic Sequential Water Maze

In this section, we extend our original task, developing a more complex paradigm called the Dynamic Sequential Water Maze (dsWM). In the previous model, an agent was positioned within five distinct environments, each containing a unique goal location. However, the increased complexity of dsWM necessitates a more advanced set of cognitive capabilities from the agent.

In this version of the task, the agent must now retain the goal position for each environment while additionally *adapting* to altered goal positions within a single environment. To elaborate, the agent is initially trained in one environment, with a fixed goal position. Following this training period, the goal position is changed twice, yet the agent's policy is not trained further. The agent is then tested a further 800 times each in two different goal positions. This procedure is subsequently repeated in four additional unique environments, without the initial training period.

In our primary study, we had compared our approach to several baseline models. These baseline models involved training the policy in one environment with a fixed goal position, followed by the relocation of the goal position within the same environment for testing. The results demonstrated that our method was unique in its ability to generalize without further training, while the baseline methods exhibited poor performance.

The Dynamic Sequential Morris Water Maze represents an extension of this original work, offering a more complex task and demanding greater cognitive adaptability from the agent. This enhanced task complexity will allow us to analyze the capability of our method further. Figure 6 shows that our MSN is robust against all inter- and intra-environment changes.

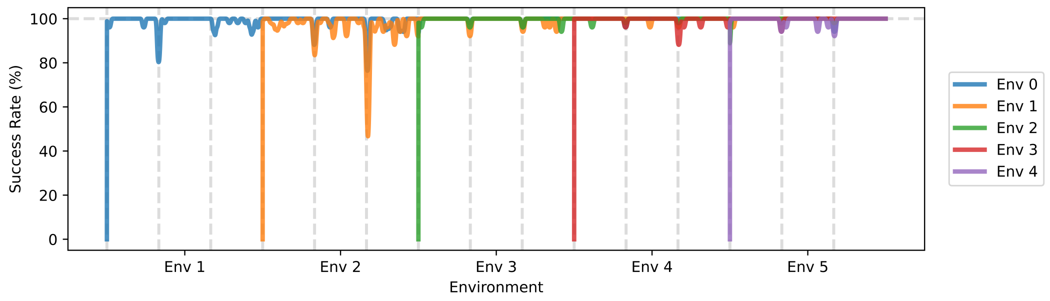

Figure 5: Our method shows robustness against all inter- and intra- environment changes. The average success rate of each environment while training on the following environments of our final model. Each dotted line indicates a goal position change. We use a moving average of 25 points and gaussian smoothing with $\sigma = 10$.

E Replay Buffer

In order to assess the sensitivity of our baseline replay buffer, we conducted tests using a variety of replay buffer sizes, mirroring our original experimental setup. Initially, the network was trained in one environment and fine-tuned using 100 randomly sampled data points from the replay buffer. Subsequent testing was performed on all previously trained environments. The data obtained from these tests reveals no correlation between replay buffer size and performance in the latter environments, suggesting catastrophic forgetting.

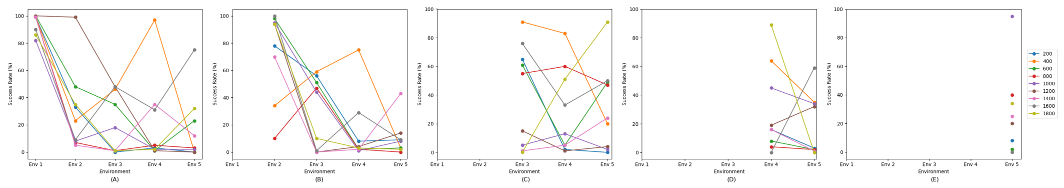

Figure 6: Each graph (A-E) depicts the model's performance accuracy in a specific environment (1-5). This accuracy is evaluated following initial training in a new environment and subsequent refinement via the replay buffer. The lines within each graph correspond to varying sizes of the replay buffer. Each data point on these lines represents the model's accuracy within the represented environment, post-training in the preceding environment.

F Computing Infrastructure

We conducted our experiments on a high-performance computing system. The system was equipped with an AMD EPYC 7713 64-Core Processor, 32 GB of RAM and a Nvidia RTX 2080 ti GPU.