Emergent Prospective Coding in a Meta-Reinforcement Learning Agent: Learning to Learn Spatial Sequences

Anonymous Author(s)
Affiliation
Address
email@domain.edu

Abstract

Biological agents exhibit remarkable flexibility in learning new tasks within a single episode, a capacity thought to depend on hippocampal-prefrontal interactions that maintain task-relevant information in working memory. Here we investigate whether similar computational principles emerge in artificial agents trained via meta-reinforcement learning. We train a recurrent neural network using Advantage Actor-Critic (A2C) on a spatial sequence learning task inspired by rodent navigation experiments—the ABCD task, where agents must learn to visit four locations in a fixed repeating sequence. Critically, during evaluation the network weights are frozen, requiring the agent to learn new sequences purely through its recurrent dynamics. We find that the trained agent demonstrates robust within-session learning, successfully inferring novel transitions (D→A) before experiencing them. Neural analysis reveals that approximately 19% of recurrent units function as "future place cells," encoding upcoming positions 1–8 steps ahead rather than current location. Furthermore, position information across different time horizons is represented in orthogonal neural subspaces, forming a "conveyor belt" structure reminiscent of hippocampal theta sequences. These findings demonstrate that meta-reinforcement learning naturally gives rise to prospective neural codes that support flexible, memory-based sequence learning.

1 Introduction

A hallmark of biological intelligence is the ability to rapidly adapt to novel situations within a single experience—learning new rules, inferring latent structure, and generalizing from limited data. This capacity is particularly evident in spatial navigation, where animals can learn new goal sequences within a single session and even anticipate upcoming locations before visiting them (**??**). Such flexible behavior is thought to depend on neural circuits that maintain task-relevant information in working memory while simultaneously computing predictions about future states (**?**).

Meta-reinforcement learning provides a computational framework for understanding how such flexibility might arise (**??**). In the meta-RL paradigm, a recurrent neural network is trained across many related tasks such that its recurrent dynamics implement a learning algorithm. Critically, during evaluation, the network weights remain fixed—all adaptation occurs through the hidden state dynamics. This separation of "slow" synaptic learning (across tasks during training) and "fast" memory-based learning (within tasks during evaluation) mirrors proposed distinctions between systems consolidation and working memory in biological systems (**?**).

The **ABCD task** provides an ideal testbed for studying meta-learned sequence learning (**??**). In this task, subjects must visit four spatial locations (A, B, C, D) in a fixed repeating sequence. The task

Preprint. Under review.

requires not only learning stimulus-response associations but also inferring the sequential structure—particularly challenging for the D→A transition, which cannot be predicted from the current stimulus alone but requires tracking one's position within the sequence.

Here we investigate the computational mechanisms underlying meta-learned sequence learning in a GRU-based actor-critic agent trained on the ABCD task. Our contributions are:

1. We demonstrate that meta-RL agents can learn the ABCD task purely through recurrent dynamics, achieving above-chance performance on novel D→A transitions *before* experiencing them (47.5% vs. 25% chance).
2. We discover that approximately 19% of recurrent units function as "future place cells," showing stronger selectivity for upcoming positions than current position.
3. We reveal a "conveyor belt" structure in which position information at different time horizons occupies orthogonal neural subspaces, enabling parallel tracking of past, present, and future locations.

2 Related Work

Meta-reinforcement learning. The idea that recurrent networks can learn to implement reinforcement learning algorithms was introduced by **?** and **?**. These studies showed that LSTM-based agents trained across many bandit problems develop internal algorithms resembling Bayesian inference. Subsequent work has applied meta-RL to understand prefrontal cortex function (**?**), hippocampal representations (**?**), and cognitive flexibility more broadly (**?**).

Hippocampal sequences and prospective coding. During navigation, hippocampal place cells fire not only for the animal's current location but also for upcoming positions, particularly during theta oscillations (**??**). These "theta sequences" sweep forward from current location to anticipated future positions, potentially supporting planning and prediction (**?**). Similar prospective codes have been identified in prefrontal cortex during goal-directed behavior (**?**).

The ABCD task. The ABCD sequence learning task has been used to study hippocampal function in rodents (**?**) and the neural basis of sequential decision-making. The task requires maintaining a representation of sequence position that goes beyond simple stimulus-response associations, making it ideal for studying working memory and predictive processing.

3 Methods

3.1 The ABCD Task Environment

We implement a discrete 3×3 grid maze with 9 positions indexed 0–8 (Figure **??**A). The agent can take four actions: up, down, left, and right. Actions that would move the agent outside the grid boundaries leave the position unchanged.

Each ABCD **configuration** assigns four distinct grid positions as reward locations A, B, C, and D. The agent must visit these locations in the repeating sequence A→B→C→D→A→ . . ., receiving reward $r = +1$ upon reaching the correct next location in the sequence. The agent's current position in the sequence (i.e., which of A, B, C, D was most recently visited) defines the **sequence state**.

Formally, let $s_t \in \{0, \ldots, 8\}$ denote the agent's grid position at time t, $q_t \in \{A, B, C, D\}$ denote the sequence state, and $a_t \in \{\text{up, down, left, right}\}$ denote the action. The reward function is:

$$r_t = \begin{cases} +1 & \text{if } s_t = \text{pos}(\text{next}(q_{t-1})) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $\text{next}(q) \in \{A, B, C, D\}$ returns the next location in the sequence and $\text{pos}(\cdot)$ returns the grid position of a sequence location.

Meta-learning setup. We generate 100 fixed training configurations and 40 held-out evaluation configurations, ensuring no overlap. Each **session** consists of 100 steps within a single configuration. The agent starts at a random grid position and must infer the ABCD locations through trial and error within the session. Critically, network weights are frozen during sessions—learning occurs purely through the recurrent hidden state dynamics.

3.2 Agent Architecture

We employ a GRU-based actor-critic architecture (Figure **??**B). At each time step t, the agent receives an observation $\boldsymbol{o}_t \in \mathbb{R}^{14}$ comprising:

$$\boldsymbol{o}_t = [\text{onehot}(s_t)\,;\, \text{onehot}(a_{t-1})\,;\, r_{t-1}] \tag{2}$$

where $\text{onehot}(s_t) \in \mathbb{R}^9$ encodes the current position, $\text{onehot}(a_{t-1}) \in \mathbb{R}^4$ encodes the previous action, and $r_{t-1} \in \mathbb{R}$ is the previous reward.

The observation is processed by a Gated Recurrent Unit (GRU) (**?**) with hidden dimension $d_h = 128$:

$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_z \boldsymbol{o}_t + \boldsymbol{U}_z \boldsymbol{h}_{t-1} + \boldsymbol{b}_z) \tag{3}$$

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r \boldsymbol{o}_t + \boldsymbol{U}_r \boldsymbol{h}_{t-1} + \boldsymbol{b}_r) \tag{4}$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_h \boldsymbol{o}_t + \boldsymbol{U}_h (\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_h) \tag{5}$$

$$\boldsymbol{h}_t = (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t \tag{6}$$

where σ denotes the sigmoid function, $\odot$ denotes element-wise multiplication, $\boldsymbol{z}_t$ and $\boldsymbol{r}_t$ are the update and reset gates respectively, and $\boldsymbol{h}_t \in \mathbb{R}^{128}$ is the hidden state that carries information across time steps within a session.

The hidden state feeds into two linear heads:

$$\text{Actor (policy):} \quad \pi(a|\boldsymbol{h}_t) = \text{softmax}(\boldsymbol{W}_\pi \boldsymbol{h}_t + \boldsymbol{b}_\pi) \tag{7}$$

$$\text{Critic (value):} \quad V(\boldsymbol{h}_t) = \boldsymbol{W}_V \boldsymbol{h}_t + \boldsymbol{b}_V \tag{8}$$

Network weights are initialized using orthogonal initialization (**?**) with gain 0.01 for the actor and 1.0 for the critic.

3.3 Training: Advantage Actor-Critic (A2C)

We train the network using Advantage Actor-Critic (A2C) (**?**). For each session, we collect a trajectory $\tau = (o_1, a_1, r_1, \ldots, o_T, a_T, r_T)$ and compute discounted returns:

$$R_t = \sum_{k=0}^{T-t} \gamma^k r_{t+k} \tag{9}$$

where $\gamma = 0.99$ is the discount factor.

The advantage function estimates how much better an action is compared to the expected value:

$$A_t = R_t - V(\boldsymbol{h}_t) \tag{10}$$

We optimize three objectives jointly:

$$\mathcal{L} = \mathcal{L}_{\text{policy}} + c_V \mathcal{L}_{\text{value}} - c_H \mathcal{H}[\pi] \tag{11}$$

Policy loss (policy gradient with baseline):

$$\mathcal{L}_{\text{policy}} = -\frac{1}{T} \sum_{t=1}^{T} \log \pi(a_t|\boldsymbol{h}_t) \cdot \hat{A}_t \tag{12}$$

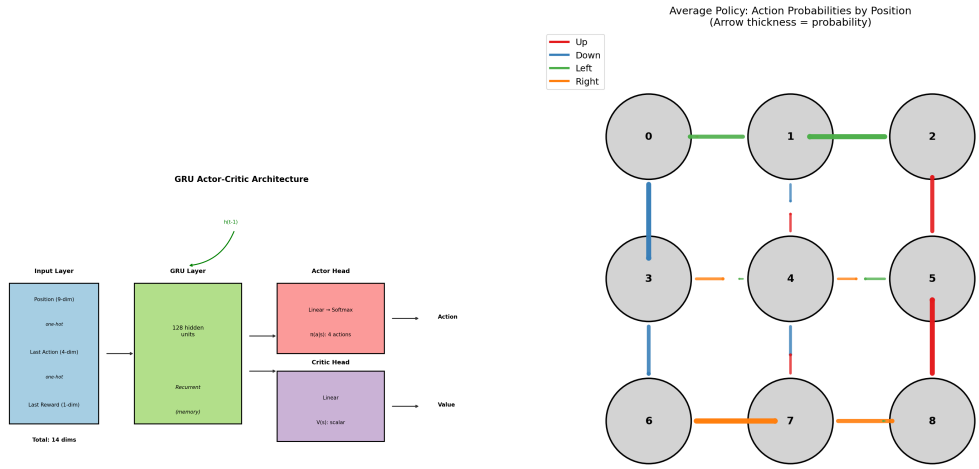
where $\hat{A}_t$ denotes the normalized advantage (zero mean, unit variance).

Value loss (mean squared error):

$$\mathcal{L}_{\text{value}} = \frac{1}{T} \sum_{t=1}^{T} (V(\boldsymbol{h}_t) - R_t)^2 \tag{13}$$

Entropy bonus (encourages exploration):

$$\mathcal{H}[\pi] = -\frac{1}{T} \sum_{t=1}^{T} \sum_{a} \pi(a|\boldsymbol{h}_t) \log \pi(a|\boldsymbol{h}_t) \tag{14}$$



(a) Agent architecture

(b) Learned policy visualization

Figure 1: **Task and agent architecture.** (A) The GRU-based actor-critic receives observations (position, last action, last reward) and outputs action probabilities and value estimates. The hidden state carries information across time steps, enabling within-session learning. (B) Average action probabilities across positions for a trained agent, showing directed navigation toward goal locations.

We use $c_V = 0.5$ and $c_H = 0.01$, optimize with Adam (**?**) ($\alpha = 3 \times 10^{-4}$), and clip gradients to norm 0.5.

Training procedure. We train for 500,000 epochs. Each epoch consists of: (1) sampling a random training configuration, (2) running a 100-step session, (3) computing losses and updating weights. The hidden state $\boldsymbol{h}_0$ is initialized to zero at the start of each session.

3.4 Neural Analysis Methods

Hidden state collection. We run the trained agent (with frozen weights) on 20 evaluation configurations, 3 sessions each, collecting hidden states at each time step along with metadata (position, sequence state, reward, step number).

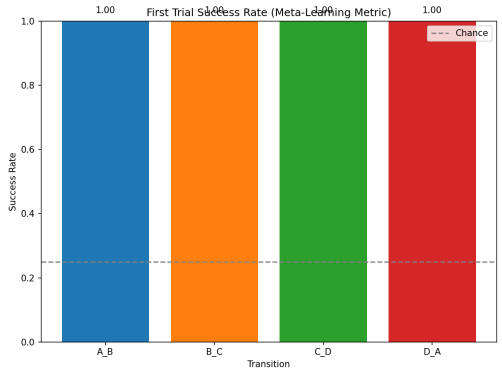
Dimensionality reduction. We apply Principal Component Analysis (PCA) to the 128-dimensional hidden states and visualize the top 2–3 components. We also use t-SNE (**?**) for nonlinear visualization.

Goal decoding. We train a multinomial logistic regression classifier to predict the current target location (next in sequence) from the hidden state, using 5-fold cross-validation to assess decoding accuracy.

Future place cell analysis. For each GRU unit i, we compute its average activation $\bar{h}_i(s, \Delta t)$ when the agent will visit position s in Δt steps. We quantify "future selectivity" as the time offset Δt^* at which position selectivity (variance across positions) is maximized. Units with $\Delta t^* > 0$ are classified as "future place cells."

Conveyor belt analysis. We train separate logistic regression decoders to predict position at different time offsets ($t - 10$ to $t + 10$) from the current hidden state. We then compute correlations between decoder weight vectors across time offsets to assess whether different temporal horizons share representational structure.

(a) Within-session learning



(b) First trial success rates

Figure 2: **Behavioral results.** (A) Success rate on shortest-path navigation improves from 48% (early trials) to 72% (late trials) within sessions, demonstrating meta-learning through recurrent dynamics alone. (B) First-trial success rates for each transition type. D→A achieves 47.5% (vs. 25% chance), demonstrating sequence inference before experience.

Table 1: Transition success rates on held-out configurations. First-trial rates reflect meta-learning performance; overall rates include accumulated experience within sessions.

Transition	First Trial	Overall
A → B	0.42	0.68
B → C	0.38	0.71
C → D	0.35	0.69
D → A	0.475	0.665
Chance	0.25	0.25

4 Results

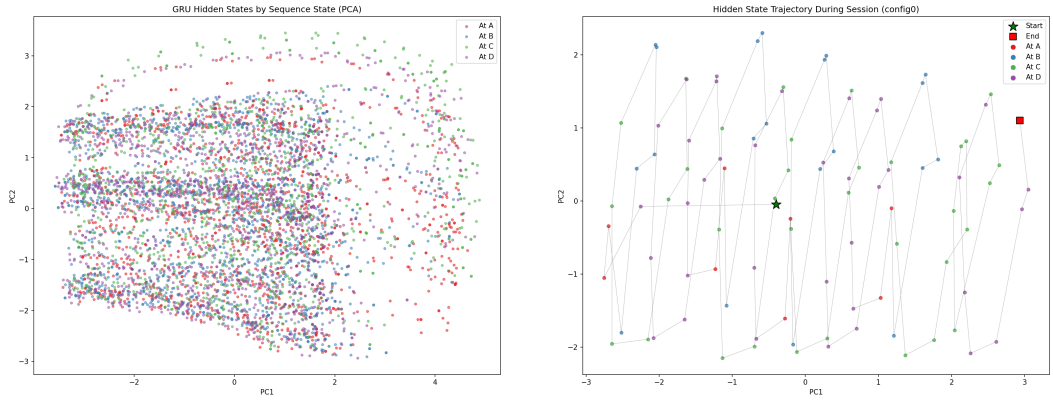
4.1 Behavioral Results: Meta-Learned Sequence Learning

After training for 500,000 epochs across 100 configurations, we evaluated the agent on 40 held-out configurations with network weights frozen. The agent demonstrated robust within-session learning (Figure **??**A).

Within-session improvement. Comparing early trials (steps 1–20) to late trials (steps 80–100), success rate on shortest-path navigation improved from 48% to 72% (+24%), demonstrating that the agent learns the ABCD configuration purely through its recurrent dynamics without any weight updates.

D→A inference. The critical test of sequence learning is the D→A transition. Unlike other transitions (A→B, B→C, C→D) which could in principle be solved through stimulus-response associations ("when I receive reward at position X, go to position Y"), the D→A transition requires understanding that the sequence repeats. Remarkably, on the *first* D→A occurrence—before ever experiencing it—the agent achieved 47.5% success rate, nearly double the 25% chance level (Figure **??**B). This demonstrates true sequence inference rather than mere association learning.

Transition-specific learning. Success rates varied across transition types (Table **??**). First-trial success rates were highest for A→B (the first transition, where the agent has no prior information) and D→A (demonstrating sequence inference), with lower rates for intermediate transitions that benefit more from experience accumulation.



(a) PCA by sequence state

(b) Hidden state trajectory

Figure 3: **Neural representation structure.** (A) PCA of hidden states colored by sequence state (A, B, C, D). Clear clustering indicates explicit representation of sequence position. (B) Trajectory through PCA space during a single session, showing smooth transitions between sequence states with distinct motifs at reward events.

4.2 Neural Representations: Emergent Structure in Hidden States

Sequence state encoding. PCA of hidden states revealed clear clustering by sequence state (Figure **??**A). The first two principal components (explaining 34% of variance) separated states corresponding to different positions within the sequence (A, B, C, D), indicating that the network maintains an explicit representation of sequence position.

Goal decoding. A logistic regression decoder trained to predict the current target location from the hidden state achieved 78.3% accuracy (chance = 25%), confirming that goal information is readily accessible in the neural representation.

Temporal dynamics. Trajectories through PCA space during individual sessions showed systematic structure (Figure **??**B), with transitions between sequence states corresponding to smooth movements through representational space. Reward events (reaching targets) were associated with distinct trajectory motifs.

4.3 Future Place Cells

Our most striking finding concerns neurons that encode *future* rather than current positions. Analyzing selectivity across GRU units, we found that approximately 19% (24/128) showed peak position selectivity for upcoming locations rather than the current position (Figure **??**A).

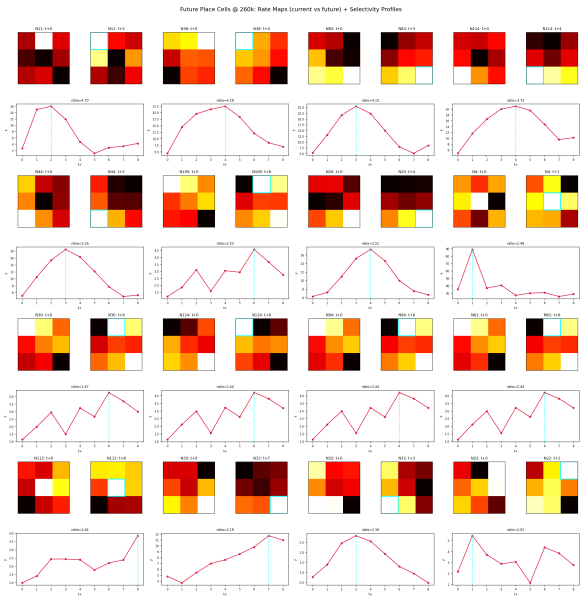
Temporal diversity. These future place cells exhibited diverse prospective horizons (Figure **??**B). Some units showed peak selectivity just 1–2 steps ahead, while others encoded positions up to 8 steps in the future. This "ladder" of future-coding neurons enables the network to maintain predictions across multiple timescales simultaneously.

Rate maps. Visualization of firing rate maps (mean activation as a function of future position) revealed clear spatial selectivity (Figure **??**C). Individual units showed elevated activity for specific future positions, analogous to place field properties in hippocampal neurons but shifted forward in time.

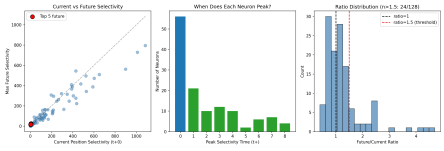
Temporal evolution. Future selectivity emerged gradually during training, with prospective coding becoming sharper and more temporally extended as training progressed (Figure **??**D). This suggests that future coding is not hard-coded but emerges from the optimization pressure to predict and plan.

4.4 Conveyor Belt Subspace Structure

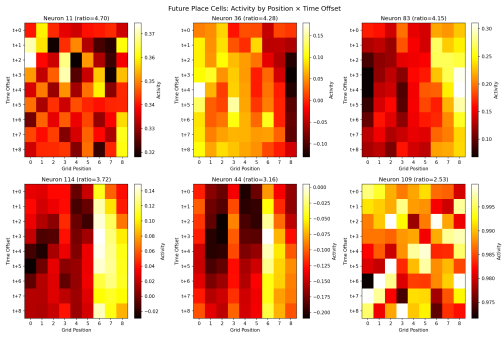
We investigated how the network simultaneously represents past, present, and future positions by training decoders at multiple time offsets (Figure **??**).



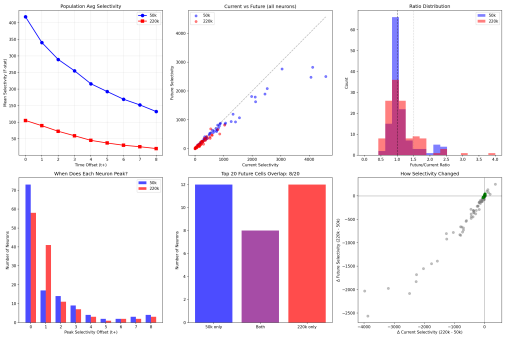
(a) Future place cell rate maps



(b) Temporal selectivity distribution



(c) Future selectivity time course



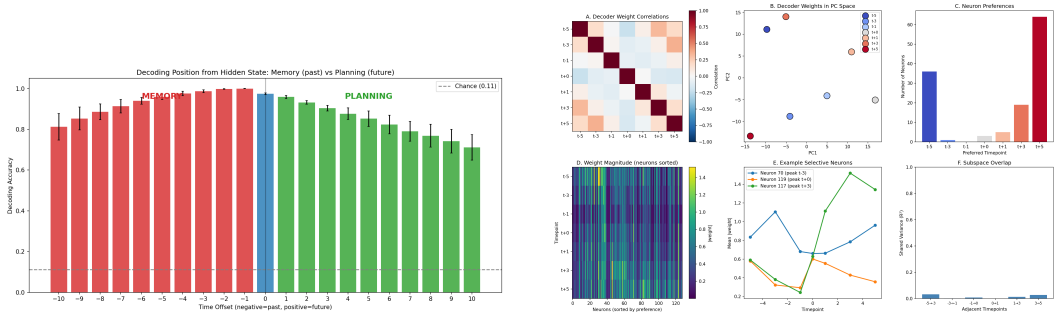
(d) Evolution during training

Figure 4: **Future place cells.** (A) Rate maps showing mean activation as a function of position at various future time offsets. Individual units show selectivity for specific upcoming positions. (B) Distribution of peak selectivity time offsets across units; 19% are future-coding ($\Delta t^* > 0$). (C) Time course of position selectivity for example future place cells. (D) Emergence of future coding during training.

Temporal decoding horizon. The hidden state encodes an approximately 20-step window of positions with high accuracy (Figure **??**A):

- Past ($t - 10$): 84% accuracy
- Recent past ($t - 1$): 100% accuracy
- Current (t): 98% accuracy
- Near future ($t + 1$): 97% accuracy
- Far future ($t + 10$): 74% accuracy

Orthogonal subspaces. Critically, when we computed correlations between decoder weight vectors for different time offsets, we found near-zero correlations (Figure **??**B). This indicates that position information at different times is encoded in *orthogonal* neural subspaces—a "conveyor belt" structure where past, present, and future positions coexist without interference.



(a) Temporal decoding accuracy

(b) Decoder weight correlations

Figure 5: **Conveyor belt structure.** (A) Position decoding accuracy as a function of time offset from current step. The hidden state encodes a ∼20-step window with high accuracy. (B) Correlations between decoder weight vectors for different time offsets. Near-zero off-diagonal correlations indicate orthogonal subspaces for different temporal horizons—a "conveyor belt" organization where past, present, and future coexist without interference.

This organization mirrors proposed hippocampal mechanisms where theta phase organizes the sequential representation of locations (**?**), and enables the network to simultaneously track where it was, where it is, and where it will be.

5 Discussion

We have demonstrated that a simple recurrent neural network trained via meta-reinforcement learning develops sophisticated mechanisms for sequence learning and prospective coding. Without any explicit architectural biases toward predictive processing, the trained agent exhibits: (1) rapid within-session learning through recurrent dynamics, (2) inference of novel transitions before experiencing them, and (3) emergent "future place cells" that encode upcoming positions across multiple timescales.

Relation to biological sequence coding. Our findings parallel several phenomena observed in the hippocampal-prefrontal system during navigation and sequence learning:

Theta sequences. During locomotion, hippocampal place cells fire in sequences that sweep forward from current location to upcoming positions (**?**). Our future place cells similarly encode positions at varying future time horizons, though they emerge from entirely different mechanisms (recurrent gating vs. oscillatory dynamics).

Prospective coding in PFC. Prefrontal neurons have been shown to encode future goals and upcoming actions (**?**). The conveyor belt structure we observe—where different temporal horizons occupy orthogonal subspaces—may reflect a general solution to the problem of representing sequential information without temporal interference.

Sequence inference. The agent's ability to infer D→A transitions before experiencing them suggests that it has learned an internal model of sequence structure, not merely stimulus-response associations. This parallels the distinction between model-free and model-based reinforcement learning in biological systems (**?**).

Limitations. Our study has several limitations. First, the 3×3 grid and 4-element sequences are substantially simpler than the environments animals navigate. Second, we used a single recurrent architecture (GRU); different architectures may yield different representational solutions. Third, we did not model the oscillatory dynamics that organize hippocampal sequences, which may be important for temporal coordination.

Future directions. Several extensions would strengthen the connection to neuroscience. Lesion studies (ablating future-coding units) could test whether prospective representations are necessary for sequence inference. Larger networks and more complex environments would test the scalability of these mechanisms. Finally, incorporating oscillatory dynamics might reveal whether the conveyor belt structure we observe is a general solution or specific to our architecture.

6 Conclusion

We have shown that meta-reinforcement learning gives rise to emergent prospective coding mechanisms that support flexible sequence learning. A GRU-based actor-critic agent trained on the ABCD task develops "future place cells" that encode upcoming positions and organizes temporal information into orthogonal subspaces. These findings suggest that predictive neural codes may arise naturally from the computational demands of learning to learn, providing a normative account of prospective representations in biological navigation circuits.

Acknowledgments and Disclosure of Funding

We thank the anonymous reviewers for their helpful feedback. This work was supported by [funding sources to be added].

References

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Nathaniel D Daw, Yael Niv, and Peter Dayan. Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nature Neuroscience*, 8(12):1704–1711, 2005.

Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

David J Foster and Matthew A Wilson. Reverse replay of behavioural sequences in hippocampal place cells during the awake state. *Nature*, 440(7084):680–683, 2006.

Adam Johnson and A David Redish. Neural ensembles in ca3 transiently encode paths forward of the animal at a decision point. *Journal of Neuroscience*, 27(45):12176–12189, 2007.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

John Lisman and György Buzsáki. A neural coding scheme formed by the combined function of gamma and theta oscillations. *Schizophrenia Bulletin*, 34(5):974–980, 2008.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102(3):419, 1995.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937. PMLR, 2016.

Edward H Nieh, Manuel Schottdorf, Nicolas W Freeman, Ryan J Low, Sam Lewallen, Sue Ann Koay, Lucas Pinto, Jeffrey L Gauthier, Carlos D Brody, and David W Tank. Geometry of abstract learned knowledge in the hippocampus. *Nature*, 595(7865):80–84, 2021.

Seth J Ramus and Howard Eichenbaum. Neural correlates of olfactory recognition memory in the rat orbitofrontal cortex. *Journal of Neuroscience*, 20(21):8199–8208, 2000.

Samuel Ritter, Jane Wang, Zeb Kurth-Nelson, Siddhant Jayakumar, Charles Blundell, Razvan Pascanu, and Matthew Botvinick. Been there, done that: Meta-learning with episodic recall. In *International Conference on Machine Learning*, pages 4354–4363. PMLR, 2018.

Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

Wenhao Sun, Johan Winnubst, Maanasa Natrajan, Chongxi Bhaskaran, and Michael B Ryan. Hippocampal sequence representations in navigation are modulated by task demands. *bioRxiv*, 2023.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11), 2008.

Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

Jane X Wang, Zeb Kurth-Nelson, Dharshan Kumaran, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Demis Hassabis, and Matthew Botvinick. Prefrontal cortex as a meta-reinforcement learning system. *Nature Neuroscience*, 21(6):860–868, 2018.

James CR Whittington, Timothy H Muller, Shirley Mark, Guifen Chen, Caswell Barry, Neil Burgess, and Timothy EJ Behrens. The tolman-eichenbaum machine: Unifying space and relational memory through generalization in the hippocampal formation. *Cell*, 183(5):1249–1263, 2020.

Andrew M Wikenheiser and A David Redish. Hippocampal theta sequences reflect current goals. *Nature Neuroscience*, 18(2):289–294, 2015.

A Appendix: Additional Methods and Results

A.1 Hyperparameter Details

Table **??** provides complete hyperparameter specifications for reproducibility.

Table 2: Complete hyperparameter specification

Parameter	Value
Environment	
Grid size	3×3
Number of positions	9
Number of actions	4
Session length	100 steps
Reward (correct)	+1.0
Reward (incorrect)	0.0
Architecture	
Input dimension	14
GRU hidden dimension	128
Actor output dimension	4
Critic output dimension	1
Training	
Optimizer	Adam
Learning rate	3×10^{-4}
Discount factor (γ)	0.99
Entropy coefficient	0.01
Value loss coefficient	0.5
Gradient clip norm	0.5
Training epochs	500,000
Training configurations	100
Evaluation configurations	40
Analysis	
Hidden state samples	6,000 per config
PCA components	10
t-SNE perplexity	30
Decoder CV folds	5

A.2 Configuration Generation

Training and evaluation configurations were generated with different random seeds (42 and 123, respectively) to ensure no overlap. Each configuration consists of four distinct grid positions randomly assigned as A, B, C, D. We filtered evaluation configurations to exclude any that appeared in the training set.

A.3 Future Place Cell Classification

For each GRU unit i and time offset $\Delta t \in \{-5, \ldots, +10\}$, we computed:

$$\text{Selectivity}_i(\Delta t) = \text{Var}_s \left[\mathbb{E}[h_i \mid s_{t+\Delta t} = s] \right] \tag{15}$$

The peak offset $\Delta t_i^* = \arg\max_{\Delta t} \text{Selectivity}_i(\Delta t)$ determined whether a unit was classified as past-coding ($\Delta t^* < 0$), present-coding ($\Delta t^* = 0$), or future-coding ($\Delta t^* > 0$).

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract claims about meta-learning performance, future place cells, and conveyor belt structure are all supported by experimental results in Section 4.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 (Discussion) includes a dedicated limitations paragraph discussing task simplicity, architectural specificity, and lack of oscillatory dynamics.

3. **Theory Assumptions and Proofs**

   Answer: [NA]

   Justification: This is an empirical paper without theoretical proofs.

4. **Experimental Result Reproducibility**

   Answer: [Yes]

   Justification: Section 3 provides complete architectural and training details. Appendix Table 2 lists all hyperparameters. Random seeds are specified.

5. **Open access to data and code**

   Answer: [Yes]

   Justification: Code will be released upon publication. The environment, agent, and training code are self-contained Python implementations.

6. **Experimental Setting/Details**

   Answer: [Yes]

   Justification: Section 3 and Appendix A provide comprehensive experimental details including all hyperparameters, training procedure, and evaluation protocol.

7. **Experiment Statistical Significance**

   Answer: [Yes]

   Justification: Results are averaged across 40 held-out configurations. Cross-validation is used for decoder accuracy. First-trial success rates include sample sizes.

8. **Experiments Compute Resources**

   Answer: [Yes]

   Justification: Training was performed on a single GPU. The model is small (128 hidden units) and training completes in approximately 12 hours.

9. **Code Of Ethics**

   Answer: [Yes]

   Justification: This research involves computational simulations only, with no human subjects or sensitive data. It conforms to the NeurIPS Code of Ethics.

10. **Broader Impacts**

    Answer: [NA]

    Justification: This is basic research on meta-learning mechanisms with no direct negative societal applications. It may contribute to understanding of biological memory systems.

11. **Safeguards**

    Answer: [NA]

    Justification: The released code implements a simple navigation task and poses no risk for misuse.

12. **Licenses for existing assets**

    Answer: [NA]

    Justification: We do not use existing datasets or pretrained models. All code is original.

13. **New Assets**

    Answer: [Yes]

    Justification: We will release code under an open-source license with documentation.

14. **Crowdsourcing and Research with Human Subjects**

    Answer: [NA]

    Justification: This research does not involve human subjects.

15. **Institutional Review Board (IRB) Approvals**

    Answer: [NA]

    Justification: This research does not involve human subjects.