A Biologically Interpretable Cognitive Architecture for Online Structuring of Episodic Memories into Cognitive Maps

Evgenii Dzhivelikian[1], **Nikita Bainaev-Mangilev**[2], **Aleksandr Panov**[1, 2]

[1]Cognitive AI Systems Lab
[2]Moscow Independent Research Institute of Artificial Intelligence
dzhivelikian@cogailab.com, bainaevn@miriai.org, panov@cogailab.com

Abstract

Cognitive maps provide a powerful framework for understanding spatial and abstract reasoning in biological and artificial agents. While recent computational models link cognitive maps to hippocampal-entorhinal mechanisms, they often rely on global optimization rules (e.g., backpropagation) that lack biological plausibility. In this work, we propose a novel cognitive architecture for structuring episodic memories into cognitive maps compatible with neural substrate constraints. Our model integrates the Successor Features framework with episodic memories, enabling incremental, online learning through agent-environment interaction. We demonstrate its efficacy in a partially observable gridworld, where the architecture autonomously organizes memories into structured representations without centralized optimization. This work bridges computational neuroscience and AI, offering a biologically grounded approach to cognitive map formation in artificial adaptive agents.

1 Introduction

A cognitive map is a concept that has proven useful in explaining the spatial reasoning abilities of animals and abstract reasoning in humans (Tolman 1948; Nadel 2013; Whittington et al. 2022). The ability of animals to plan and derive a general structure between different tasks, flexibly connect it to novel tasks and environments, is commonly attributed to cognitive maps. Therefore, studying neurophysiological underpinnings of cognitive maps, should bring us closer to the understanding of animal's cognition and, as a consequence, provide insights for design of more adaptable artificial agents.

Several recent studies proposed computational models that connect cognitive maps with the hippocampus and entorhinal cortex and provide an insight that cognitive maps and their neural substrate, like grid cells and place cells, could form from general machine learning computational rules, which aim to reduce the model's uncertainty about the environment (Whittington et al. 2020; George et al. 2021; Dedieu et al. 2024). As evidence suggests, cognitive maps may be a general solution for attempts of an intelligent system to structure knowledge in order to reuse it more efficiently. In order to learn these structures, computational models usually rely on a common backpropagation algorithm in artificial neural networks or on the Expectation-Maximization algorithm, as described in George et al. (2021).

It's widely discussed that backpropagation might not be supported by local neuronal interaction in the brain (Lillicrap et al. 2020) due to numerous constraints. At the same time, computational models that are based solely on Hebbian-like learning can't compete with the state of the art of ANN learning methods in their generalisation abilities and universality. Therefore, the exact mechanisms that allow for flexible online, iterative learning of generalised structures in the brain are still elusive.

Exploring alternative means for generalisation, that could be implemented in the brain, we propose a model of gradual structuring of episodic memories into cognitive maps. In contrast to many classical learning algorithms, the proposed model is inherently agentic: it is based on the Successor Features (SF) framework (Barreto et al. 2018), episodic memories and requires interactions with the environment in order to form structured knowledge. Experiments in a partially observable grid-world environment show how memories can be structured incrementally in a fully online fashion with local Hebbian-like rules within our agent architecture, bridging the gap between artificial intelligence and neuroscience models.

Our key contributions as follows:

- We show that unstructured episodic memories can be used to form SFs in grid-like environments.
- We testify that, under mild conditions, those SFs can be used to structure memories into clusters, semantically corresponding to the ground true environmental states.
- Based on these results, a novel biologically interpretable algorithm for hidden state structure learning was proposed.

2 Preliminaries

Let's consider an agent interacting with a partially observable environment (POE), which can be formally described as in definition 1.

Definition 1 (Partially Observable Environment)**.** Given a state space S, action space A, an observation space O, a transition function $P : S \times A \to S$, and an observation

function $X : S \to O$, a partially observable environment is a tuple:

$$\mathcal{E} = \langle S, A, O, P, X \rangle$$

Additionally, here we consider only discrete S, A, O and those mappings X in which each state corresponds to a single observation state $o \in O$. An example of such an environment is a grid-world environment, where each position corresponds to a single observation (floor colour). Such an environment is partially observable if the number of floor colours is less than the number of positions. In this case, multiple positions may correspond to the same observation state; these positions will be referred to as *clones*.

In case of POE, an agent does not have access to the true state of the environment, but only observations and actions. Therefore, it may be useful for the agent to have its own representation of the state to act efficiently in the environment. We assume that, in order to do that, the agent should learn a world model.

Definition 2 (World Model)**.** A world model is defined as a tuple comprising the space of internal or hidden states H, the action space A, the observation space O, a transition function $T(h', h, a) = \Pr(h' \mid h, a)$, an emission function $E(o, h) = \Pr(o \mid h)$, and an initial state function $B(h) = \Pr(h)$:

$$M = \langle H, A, O, T, E, B \rangle, \tag{1}$$

where $h \in H$, $a \in A$, $o \in O$ and $\Pr(\cdot)$ is a probability function.

It should be noted that, in general, $H \neq S$ and there is no bijective mapping from H to S. We consider precisely this case as the most realistic model of a cognitive agent. Therefore, we will refer to environment state $s \in S$ as the *true state*, while $h \in H$ is the agent's representation of this state, which we will call *hidden state*.

This formulation of the world model corresponds to a Hidden Markov Model (HMM), which consists of two types of random categorical variables: observation variables O_t and hidden variables H_t for each discrete time-step t. For a process of length T time steps with values of the random variables $o_{1:\mathrm{T}} = (o_1, \ldots, o_\mathrm{T})$, $h_{1:\mathrm{T}} = (h_1, \ldots, h_\mathrm{T})$ and actions $a_{1:\mathrm{T}} = (a_1, a_2, ..., a_\mathrm{T})$, the Markov property yields the following factorization of the generative model:

$$\Pr(o_{1:\mathrm{T}}, h_{1:\mathrm{T}} \mid a_{1:\mathrm{T}}) = B(h_1) \prod_{t=2}^{\mathrm{T}} T(h_t, h_{t-1}, a_{t-1}) \prod_{t=1}^{\mathrm{T}} E(o_t, h_t). \tag{2}$$

We require that the agent's world model should accurately predict the outcome of interacting with the environment. The model describes $\mathcal{E}$ better when the surprise associated with the observation sequence is lower for any arbitrary sequence of actions. Thus, the quality of the agent's world model can be assessed by the expected surprise of the observations the agent receives while interacting with the environment:

$$\mathrm{sur}(\mathcal{E}, M_\mathrm{w}) = \mathbb{E}_{o_{1:\mathrm{T}}, a_{1:\mathrm{T}}} \left[-\log \sum_{h_{1:\mathrm{T}}} \Pr(o_{1:\mathrm{T}}, h_{1:\mathrm{T}} \mid a_{1:\mathrm{T}}) \right], \tag{3}$$

where the observation sequence $o_{1:\mathrm{T}}$ is sampled from the environment $\mathcal{E}$ following an arbitrary action sequence $a_{1:\mathrm{T}} = (a_1, a_2, ..., a_\mathrm{T})$.

3 Method

Rationale

Let's consider an HMM that maximises likelihood for each given observation sequence $o_{1:\mathrm{T}}$. The maximum likelihood is reached when observation sequences are uniquely encoded by a sequence of hidden states $h_{1:\mathrm{T}}$ and the transition matrix is deterministic. Similarly to an ideal episodic memory, such a model would perfectly store each sequence without any information loss, however, it doesn't allow for generalisations. I.e. any new sampled sequence, very likely, will have low likelihood under the model. We use such perfectly storing HMM as the first level of our cognitive architecture, which, effectively, models hippocampal episodic memory.

To get from episodic memory, to structured knowledge, we use similar idea as in Best-first Model Merging (Stolcke and Omohundro 1994). In the work, authors show that iteratively merging unique hidden states, while controlling data's likelihood under the HMM, one get an HMM that is able to generalise. One of the limitations of this method is that it requires to recompute data's likelihood for each merge candidate pair, which is inefficient. Another shortcoming is that merges within the same HMM result in irreversible losses of initial information about sequences.

Based on the idea of hidden state merging, we introduce the second level in our architecture, which represents higher-level states, that connect firs-level states, organising them into clusters. Mathematically, the second level is equivalent to the first-level HMM with merged states, where merged states are connected to the same second-level state, which we will also call a cluster, since it corresponds to a set of first-level states. The important difference is that by separating merged and the original perfect storing HMM, we ensure that there is no information loss, and we always can separate states back if needed by disconnecting first-level states from their second-level counterparts. That is, even if the second level is failed to generalise properly, we always have the first-level model to back up from. This architectural design is motivated by biological plausibility as well, since it renders the merging process as synaptic learning.

Another critical component of the model is a mechanism that allows for correct connection of first-level states to second-level states. We denote this process as clusterisation. Correct clusterisation means that only those first-level states $h^{(1)} \in H^{(1)}$ that correspond to the same true state $s \in S$ are connected to the same second-level state $h^{(2)} \in H^{(2)}$. To understand this better, let's assume that each first-level state $h^{(1)}$ has a ground true label $s \in S$, like gridworld position, in which it was formed to store an observation in a sequence. To adequately describe the environment, we need the second level to represent those positions and, therefore, the algorithm should connect first-level states with the same label to the same second-level state.

Since recomputing whole data likelihood for a model is inefficient, we propose to use a cheaper successor features

computation to increase chances for correct merge candidates in comparison to random merge pairs. The successor features representation for a given hidden state h_t (by analogy with the Successor Features described in (Barreto et al. 2018)) is a discounted sum of future observation distributions under the agent's policy π:

$$\mathrm{SF}^{\pi}_{t+T}(o = j \mid h_t) = \mathbb{E}_{a_{0:\mathrm{T}} \sim \pi} \sum_{l=0}^{\mathrm{T}} \gamma^l \Pr(o_{t+l+1} = j \mid h_t, a_{0:\mathrm{T}}), \quad (4)$$

$$\Pr(o_{t+l+1} = j \mid h_t, a_{0:\mathrm{T}}) = \sum_{h_{t+1:t+l+1}} \Pr(o_{t+l+1} = j \mid h_{t+l+1}) \prod_{\tau=t+1}^{t+l+1} \Pr(h_\tau \mid h_{\tau-1}, a_{\tau-1}), \quad (5)$$

where $\gamma \in (0, 1)$.

In this work, we propose to use SF representations for matching merge pairs. The idea is based on the intuition that hidden states that correspond to identical true states will have similar distributions of future observations, and consequently, their SF representations should also be similar. In degenerate case, if we set deterministic policy π, SF will always be the same for correct merge pairs.

We generate SF, using episodic memory that is formed as described in Algorithm 1.

Algorithm 1 Episodic memory learning

Input: o_{t+1}, a_t
$h_{t+1} \leftarrow T(h_t, a_t)$
$o^*_{t+1} \leftarrow E(h_{t+1})$
if h_{t+1} is null **or** o^*_{t+1} is not o_{t+1} **then**
  $h_{t+1} \leftarrow N + 1$ # N is the total number of hidden states
  $N \leftarrow N + 1$
  $T(h_t, a_t) \leftarrow h_{t+1}$
  $E(h_{t+1}) \leftarrow o_{t+1}$
end if
$h_t \leftarrow h_{t+1}$

In this case, the transition probability function T and emission function E are reduced to mappings. To predict the next state, we just have to look up T for the current state h_t, but if there is no entry for h_t or the prediction does not match the observed state o_{t+1}, then the new state is formed. To avoid collisions, the algorithm makes sure that this state hasn't been chosen before, therefore a state counter N is used. SF formation using episodic memory is described in Algorithm 2.

Algorithm 2 SF formation using episodic memory

Input: IS, $\gamma \in (0, 1)$, T # IS is the initial set of hidden states
Output: SF
SF ← array of zeros
PS ← IS # PS is the set of all states (nodes) on the current BFS depth
for $l = 1..\mathrm{T}$ **do**
  PS ← $\bigcup_{a \in A} \{T(h, a)\}_{h \in \mathrm{PS}}$ # get next depth nodes assuming the policy is uniform
  counts ← array of zeros
  for all $h \in$ PS **do**
    $o \leftarrow E(h)$
    counts$_o$ = counts$_o$ + 1
  end for
  $\Pr(o_{t+l+1})$ = NORMALIZE(counts)
  SF = SF + γ^{l-1} $\Pr(o_{t+l+1})$
end for

However, it is important to note that, for an arbitrary policy, the SFs generated using episodic memory HMM differ from the true SF representations derived from the environment's ground true transition matrix. Indeed, since episodic memory stores trajectories independently, it correctly predicts future observations only for a specific action sequence, starting from a hidden state h, which corresponds to a specific episode of interaction with the environment.

It can be shown, however, that to improve SF the representation generated by episodic memory, we can average it over a cluster of h states that correspond to the same true state s. This is implemented in Algorithm 2 through setting the initial set of hidden states IS to this cluster. To illustrate this, we conducted experiments with an episodic memory model that stores agent trajectories $(o_1, a_1, o_2, a_2, ..., o_T, a_T)$ obtained from a gridworld environment.

The results in Figure 1 show how episodic memory generated SF changes in comparison to ground true SF with the number of hidden states in a cluster and their consistency, which we refer to as cluster purity or homogeneity. Cluster purity is the proportion of states within a cluster whose label (in this case, the position in the gridworld maze) is equal to the state s, for which the ground true SF is generated. In these experiments, SF similarity is measured in Euclidean space as $\exp(-\left\|\mathrm{SF}^e - \mathrm{SF}\right\|_2)$, where SF^e is the representation generated from episodic memory. Thus, the results on data collected by an agent with a random policy in different 10x10 gridworld environments with ten observation states (floor colours) show that the accuracy of the representations generated by episodic memory increases with both cluster size and its purity.

We also tested whether episodic memory generated SF can be used to match hidden states to environmental states s. In order to do that, we evaluated the accuracy of hidden state cluster merging in the gridworld environment (see Algorithm 3). For each position, two clusters are formed: a probe cluster and a candidate cluster, with predefined size and purity. For each probe cluster, a classification task is solved:

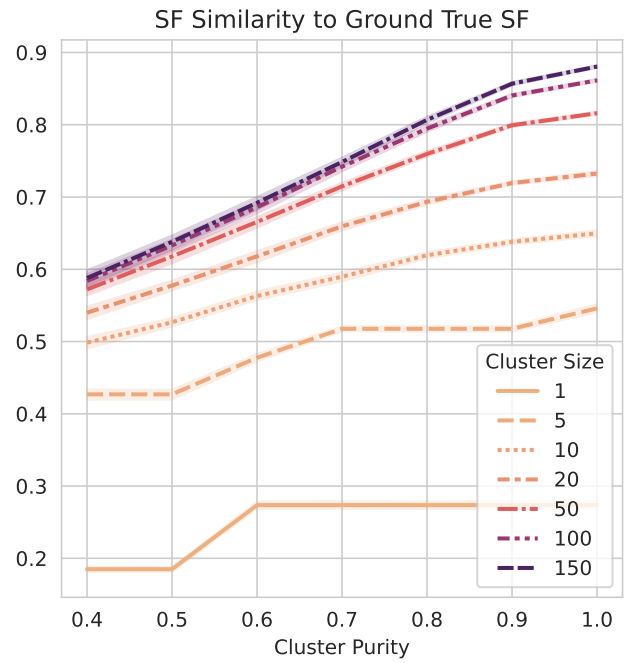

Figure 1: Dependence of the similarity between episodic memory SFs and the true SFs on the size and purity of the first-level state cluster. Results are averaged over five state partitions and three 10x10 gridworld environments with 10 colours and random colouring. The coloured shading corresponds to the 95% confidence interval.

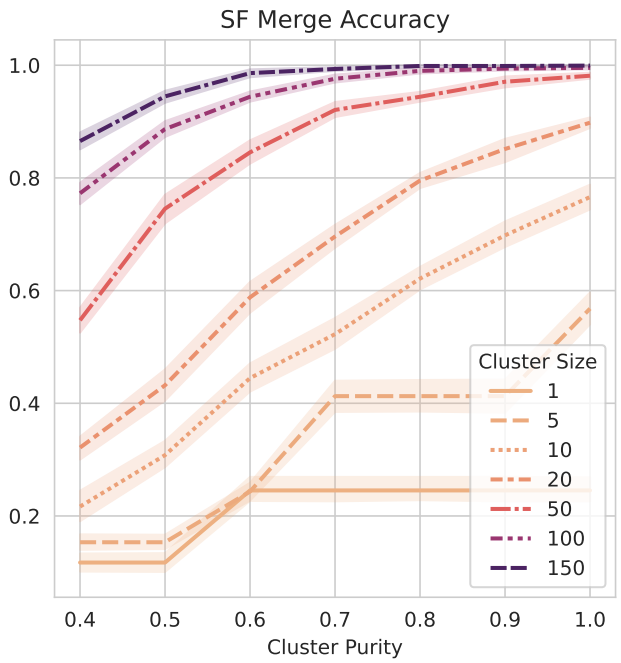

Figure 2: Dependence of the accuracy of cluster merging based on SFs on their size and purity. Results are averaged over five state partitions and three 10x10 gridworld environments with 10 colours and random colouring. The coloured shading corresponds to the 95% confidence interval.

among the candidate clusters formed based on matching observation states, only one has the same label as the probe cluster. The cluster label is defined as the mode of the first-level state labels. As can be seen from the plot in Figure 2, even for one-state clusters, the probability of correct merge is significantly above the chance, which is equal to 0.1 for 10x10 grid with 10 colours, and it quickly grows with the number of correct states in the cluster. Therefore, starting with one-state clusters and iteratively merging them into bigger clusters based on SF representations might be a good strategy.

The proposed cluster merging procedure can be described as presented in Algorithm 3. For each first-level state cluster, an SF representation is formed according to Algorithm 2, where the initial set of states includes all states in the cluster. Thus, the SF is formed by considering the superposition of future observations for all trajectories passing through the cluster's states. The clusters are then divided into two groups: probes and candidates. For each probe cluster, the similarity of its representation to every candidate cluster is computed, and the candidate cluster with the highest similarity to the probe is selected. To reduce the probability of false mergers, a threshold is applied based on how much the maximum similarity value exceeds the mean similarity, taking the standard deviation into account. Thus, if the most similar candidate cluster does not significantly deviate from the normal distribution of similarities for the probe cluster, it is less likely to be included in the list of pairs for merging. Additionally, it is reasonable to set a minimum similarity threshold below which merging is impossible.

Memory Model and its Neural Implementation

Based on hidden state cluster merging algorithm described in section 3, we propose a memory model that consists of two levels. The first level models episodic memory and is implemented as an HMM with deterministic transition matrix with maximum data likelihood (see Algorithm 1). Let us denote it as $T^{(1)} \in \{0,1\}^{n\times n}$, where n is the number of first-level states.

Let us also define the connection matrix C between the first and second-level states, of size $n \times k$, such that $C_{ij} = \Pr(h^2 = j \mid h^1 = i)$, where k is the number of second-level states, and h^1, h^2 are the hidden states of the first and second level, respectively. It can be shown that any second-level transition matrix $T^{(2)}$, obtained by merging first-level states, can be defined via the connection matrix C:

$$T^{(2)} \propto C^T \cdot (T^{(1)})^T \cdot C \tag{6}$$

Thus, merging first-level states is equivalent to having the corresponding rows in the binary matrix $C \in \{0,1\}^{n\times k}$ share the same non-zero column. This formulation also allows generalising the merging process to the case where C is a real-valued matrix. This memory model, represented as a factor graph, is shown in Figure 3.

Algorithm 3 Merging of first-level state clusters

Input: C, emb, l # list of state clusters, their SF embeddings, and merge threshold
Output: P # pairs of clusters to be merged

C_x, C_y, emb_x, emb_y ← SPLIT_SET(C, emb) # split the list of clusters and their embeddings into two parts, $\text{emb}_x \in \mathbb{R}^{n\times d}, \text{emb}_y \in \mathbb{R}^{k\times d}$
sim ← PAIRWISE_SIM(emb_x, emb_y) # pairwise similarity matrix for the two sets of clusters, $\text{sim} \in \mathbb{R}^{n\times k}$
argmax, max, mean, std ← ROWWISE_STATS(sim) # row-wise maximum, mean, and standard deviation of similarity values $\text{max}, \text{mean}, \text{std} \in \mathbb{R}^n, \text{argmax} \in \mathbb{Z}^{+n}$
$p_f = \Phi((\text{max} - \text{mean})/\text{std})$ # probability of accepting the pair with maximum similarity $p_f \in \mathbb{R}^n$, Φ is the normal CDF
$P \leftarrow \emptyset$
for i=1..n **do**
  f ← SAMPLE($p_f[i]$) # sample a Bernoulli random variable, $f \in \{0, 1\}$
  if $f = 1$ **and** $\max[i] > l$ **then**
    $P \leftarrow P \cup (C_x[i], C_y[\text{argmax}[i]])$
  end if
end for

Within this model, learning at the second level reduces to updating the connections C. Mathematically, this involves adding the corresponding columns of the matrix C for the pairs of clusters (second-level states) obtained by Algorithm 3 and zeroing out one of these columns.

A biologically plausible neural implementation of Algorithm 3 could be based on competition between groups of second-level memory neurons, whose receptive fields recognize the SF representations of the corresponding first-level state clusters. Meanwhile, the outgoing connections of these neurons should correspond to the matrix C. Competition via inhibitory interneurons should be arranged such that if several second-level neurons are active, only the synapses of the most active neuron are updated according to Hebbian rule. As shown in Figure 3, the merging phase can be divided into three stages, corresponding to the direction of signal propagation:

1. Activation of a second-level neuron $\underset{C}{\rightarrow}$ excitation of the corresponding first-level neuron cluster, $\underset{T^{(1)}}{\rightarrow}$ generation of an SF representation via recurrent connections.
2. Excitation of second-level neurons responsive to this SF representation $\underset{C}{\rightarrow}$ activation of the corresponding first-level clusters $\underset{T^{(1)}}{\rightarrow}$ update of the SF representation.
3. Winner-take-all inhibition of second-level neurons. The first level strengthens connections with the winner, and the winner strengthens connections with the neurons of the current SF representation.

Thus, to compute SF representation similarity in a neural implementation, additional weights W can be introduced. In this case, implementing a similarity metric based on cosine distance is simplest, as it is computed via the dot product. Then, the matrix W for each second-level neuron should correspond to the SF representation of the first-level state cluster associated with it.

Similar connectivity motifs can be observed in the layered structure of the mouse neocortex (Staiger and Petersen 2021). However, establishing a detailed correspondence between the proposed model and brain structures is beyond the scope of this work and requires separate consideration.

4 Experiments and Results

This section presents the results of experiments conducted in a 10x10 gridworld environment with uniform colouring (each colour appears equally often) using 10 colours and an agent performing random walks. At each step, the agent randomly selects one of four actions (up, down, left, right) and observes the floor colour at the current position, encoded as an integer. Interaction with the environment is divided into episodes of 50 action steps; upon completion, the agent is reset to the starting position (bottom-left corner).

During interaction with the environment, the agent predicts the next observation using both the first and second memory levels, but learning based on the prediction error occurs only at the first level. Every 10 episodes, the second memory level is updated by forming and merging first-level state clusters, as described in Section 3. To form SFs we set $\gamma = 0.8, T = 25$ for all experiments in Algorithm 2. Prediction accuracy can be used to assess the agent's generalization ability, as the probability of repeating a random trajectory of length 50 is very low. For comparison, the prediction accuracy of a naive first-order memory (`first order`), where observations are used as hidden states, was evaluated.

The main experimental results are presented in Figures 4, 5, and 6. The results are smoothed using a Gaussian kernel with $\sigma = 20$ and averaged over five initial random generator seeds and three random environment colourings. The coloured shading denotes a confidence interval of one standard deviation. The experiments can be divided into three groups. The first group (`random`) shows results for memory with random cluster mergers, the second (`sf`) for mergers based on SF representations, and the third (`no merge`) for no mergers at all. Within each group, three experiments were conducted with different initial cluster sizes (`size`). The experiment labelled `size: 10` means that all states are randomly partitioned into clusters of size 10 before merging. Thus, the accuracy in the `size: 1 (no merge)` experiment corresponds to the first-level memory accuracy.

Notably, partitioning into small clusters of size 10, even without merging, significantly increases prediction accuracy compared to the first level and the naive model. Indeed, even random partitions can yield sufficiently pure clusters (see Figure 5), on average increasing the generalization capability of the second-level memory. However, if the initial clusters are too large, the predictions become no better than those of the naive model.

It can also be seen that prediction accuracy is significantly higher for the group of experiments using SF representations. Initial partitioning into clusters increases the

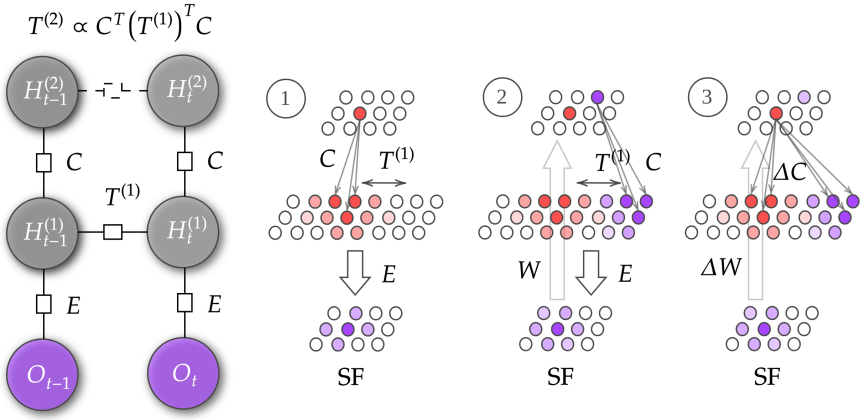

Figure 3: Factor graph of the memory model with mergers and a possible neural implementation of the merging process.

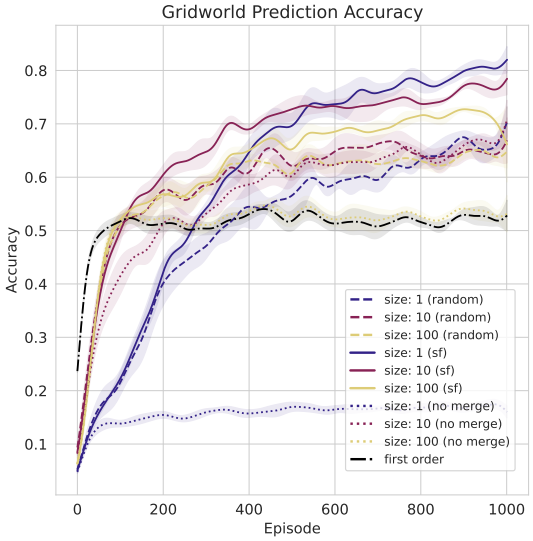

Figure 4: Mean prediction accuracy for observations as a function of the episode number in a 10x10 gridworld environment with 10 colours. The first group (`random`) shows accuracy for memory with random cluster mergers, the second (`sf`) for mergers based on SF representations, and the third (`no merge`) for no mergers. Within each group, three experiments were conducted with different initial cluster sizes (`size`). Results are smoothed with a Gaussian kernel ($\sigma = 20$) and averaged over five random seeds and three random environment colourings. The shaded area represents a one standard deviation confidence interval.

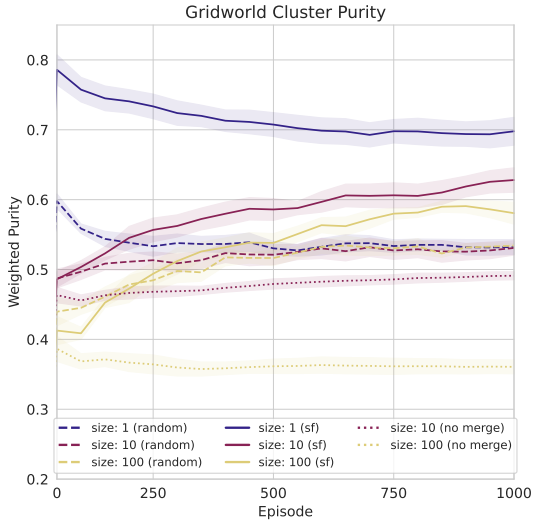

Figure 5: Weighted cluster purity as a function of the episode number in a 10x10 gridworld environment with 10 colours. The first group (`random`) is for memory with random cluster mergers, the second (`sf`) for mergers based on SF representations, and the third (`no merge`) for no mergers. Within each group, there are three experiments with different initial cluster sizes (`size`). Results are smoothed with a Gaussian kernel ($\sigma = 20$) and averaged over five random seeds and three random environment colourings. The shaded area represents a one standard deviation confidence interval.

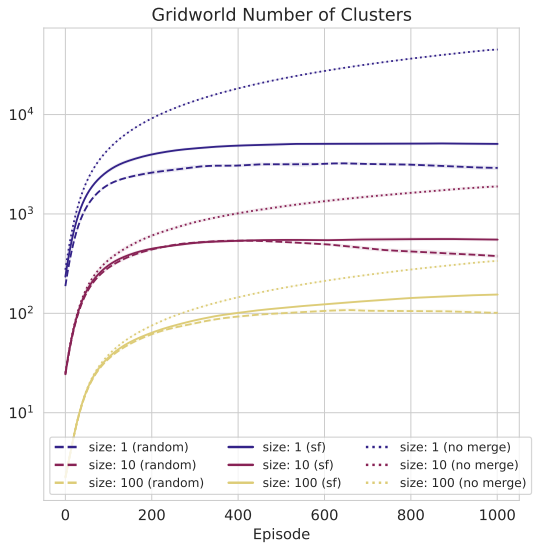

Figure 6: Number of clusters (states at the second level) as a function of the episode number in a 10x10 gridworld environment with 10 colours. The first group (`random`) is for memory with random cluster mergers, the second (`sf`) for mergers based on SF representations, and the third (`no merge`) for no mergers. Within each group, there are three experiments with different initial cluster sizes (`size`). Results are smoothed with a Gaussian kernel ($\sigma = 20$) and averaged over five random seeds and three random environment colourings. The shaded area represents a one standard deviation confidence interval.

learning rate; however, the final accuracy is higher without pre-partitioning. A possible explanation for this effect is that the initial random partition inevitably introduces noise into the second-level predictions, whereas using SFs for each merging step reduces the probability of false mergers, which can improve cluster purity. Indeed, as can be seen from the graphs in Figure 5, the highest weighted cluster purity (average homogeneity weighted by cluster size) is observed in the case without pre-partitioning.

Figure 6 also shows the growth in the number of states (clusters) at the second hierarchy level. When mergers are used, the number of states quickly stabilises, whereas without mergers, it constantly grows. It is also evident that using an initial random cluster partition significantly reduces the asymptotic number of states, increasing the computational efficiency of the memory model.

To verify that the second-level memory state space indeed reflects the environment's structure, for each experimental group (with an initial cluster partition of size 10), the second-level memory was transformed into a transition matrix between environment states (see Figure 7). For this, an algorithm analogous to the algorithm for transforming first-level transitions into a second-level matrix was used (see Equation (6)). In this case, however, the second-level states were grouped by their corresponding position labels in the environment. The label of a second-level state is defined as the mode of the labels of its constituent first-level states, which are assumed to be known for visualisation purposes. As can be seen from the visualisations, the transition matrix obtained for SF-based mergers has less pronounced off-diagonal elements, which are absent in the true transition matrix. Thus, the better prediction quality indeed correlates with a more accurate representation of the environment's transition structure.

Additionally, we compared our memory model with two other biologically interpretable online algorithms: E-prop RNN (`eprop`) (Bellec et al. 2020) and tPCN (`tpcn`) (Tang, Barron, and Bogacz 2023) as shown in Figure 8. For these experiments, we set initial cluster size to 1 and other parameters are the same as for previous experiments, except merging phase is two times more frequent. We test algorithms in the same 10x10 gridworld environment, randomly coloured with 10 colours. However, we increase the number of steps within an episode to 100 action steps, and the initial starting point is set to the central $(5, 5)$ position. This modification ensures that gridworld positions are more evenly visited. It can be seen that our proposed algorithm achieves better prediction accuracy, while requiring much less experience. All baselines were trained until convergence on the same five random gridworld colourings: E-prop RNN for 5000 episodes, tPCN—2000 episodes, and our model required only 1000 episodes to converge. Percentiles from 25th to 75th for average accuracy during last 50 episodes over five different environments are reported in Figure 8.

One of the baseline algorithms is the eligibility propagation algorithm for recurrent neural networks with spiking neurons (E-prop RNN), which was derived from the method proposed in Bellec et al. (2020) and, similarly to Murray (2019), adapted for a standard recurrent neural network architecture featuring a single recurrent layer and continuous activation functions. Model inputs consisted of concatenated one-hot encoded vectors representing both environmental observations and executed actions. Eligibility traces were computed as exponentially decaying sums of the products of input signals and corresponding learning signals, with the latter calculated via symmetric weight matrices.

The temporal predictive coding network (tPCN) model implementation was grounded in the framework outlined by Tang, Barron, and Bogacz (2023) and adapted for a discrete environment by excluding tPCN encoder part. That is, inputs were represented by one-hot encoded observation vectors that were resized to match the dimensionality of the network's hidden state. For both baselines, learning rate is multiplied by a decay factor every update. The parameters of the baselines are summarised in Table 1.

5 Conclusion

In this work, an algorithm for structuring episodic memory was proposed. It can also serve as a basis for a neurophysiological model due to its biological interpretability, as shown in Section 3. The first memory level uses a model based on the infinite-capacity HMM, modelling episodic memory. The second level is constructed by merging first-level states into clusters based on the similarity of their SF representations, which can be interpreted as the formation of connec-

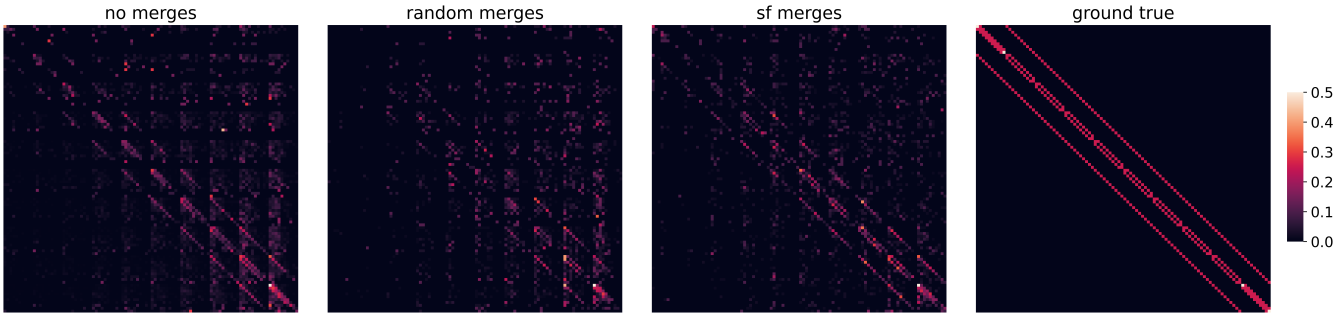

Figure 7: Transition matrices between environment positions (averaged over actions) formed based on the second-level memory under different learning regimes: `no merges` – without cluster merging, `random merges` – random mergers, `sf merges` – mergers based on SF representation similarity, `ground true` – the true transition matrix. Results are averaged over three random colourings of the 10x10 environment with 10 colours.

Model	Hidden Size	Initial Learning Rate	Learning Rate Decay	Batch Size	Recurent Activation	Out Activation
E-prop RNN	256	0.0075	0.975	32	tanh	softmax
tPCN	250	0.005	0.95	64	relu	softmax

Table 1: Main hyperparameters for baselines: tPCN and E-prop RNN.

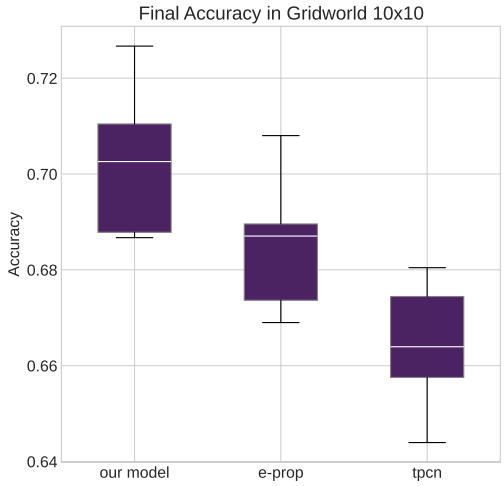

Figure 8: Prediction accuracy box plots in five random 10x10 gridworld environments with 10 colours over last 50 episodes of training. Boxes show 50% interval and span the interquartile range (IQR), from the 25th (Q1) to the 75th (Q3) percentile, while whiskers denote minimum and maximum value and the gray horizontal line corresponds to median.

tions between the first and second memory levels. In turn, SF representations may correspond to the activity patterns of place cells in the hippocampus, as discussed in Samuel J. Gershman (2018).

Experiments showed that merging states in a gridworld environment based on SF representations significantly increases the model's prediction accuracy compared to random merges and other biologically interpretable baselines like E-prop RNN and tPCN. It was also demonstrated that the increase in prediction accuracy is likely related to an improved representation of the environment's transition structure in the second-level memory.

It should be noted that within this model, it has not yet been possible to achieve the maximum prediction quality typically attained by classical algorithms (e.g., expectation–maximization or backpropagation) in similar environments. This is because the quality of mergers in the early stages of learning is low, which inevitably affects the quality of subsequent mergers. One possible solution to this problem could be using an analogue of an evolutionary algorithm for several initial random partitions, selecting the most successful ones based on prediction error. This aligns with the theory of redundancy in brain structures, where different cell ensembles duplicate each other's functions (Hawkins 2021), as well as with the theory of neuronal group selection (Edelman 1987). Another direction for developing this model could involve designing an algorithm for splitting clusters to increase their homogeneity, potentially based on classical clustering algorithms to identify a cluster's homogeneous core. Finally, adapting this algorithm for the neural implementation of a distributed version of episodic memory remains a task for future work.

References

Barreto, A.; Dabney, W.; Munos, R.; Hunt, J. J.; Schaul, T.; van Hasselt, H.; and Silver, D. 2018. Successor Features for Transfer in Reinforcement Learning. ArXiv:1606.05312 [cs].

Bellec, G.; Scherr, F.; Subramoney, A.; Hajek, E.; Salaj, D.; Legenstein, R.; and Maass, W. 2020. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications*, 11(1): 3625.

Dedieu, A.; Lehrach, W.; Zhou, G.; George, D.; and Lázaro-Gredilla, M. 2024. Learning Cognitive Maps from Transformer Representations for Efficient Planning in Partially Observed Environments. ArXiv:2401.05946 [cs].

Edelman, G. M. 1987. *Neural Darwinism: The theory of neuronal group selection*. Neural Darwinism: The theory of neuronal group selection. New York, NY, US: Basic Books. ISBN 978-0-465-04934-9. Pages: xxii, 371.

George, D.; Rikhye, R. V.; Gothoskar, N.; Guntupalli, J. S.; Dedieu, A.; and Lázaro-Gredilla, M. 2021. Clone-structured graph representations enable flexible learning and vicarious evaluation of cognitive maps. *Nature Communications*, 12(1): 2392.

Hawkins, J. 2021. *A Thousand Brains: A New Theory of Intelligence*. Basic Books. ISBN 978-1-5416-7580-3. Google-Books-ID: hYrvDwAAQBAJ.

Lillicrap, T. P.; Santoro, A.; Marris, L.; Akerman, C. J.; and Hinton, G. 2020. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6): 335–346.

Murray, J. M. 2019. Local online learning in recurrent networks with random feedback. *eLife*, 8: e43299.

Nadel, L. 2013. *Cognitive maps.*, 155–171. Handbook of spatial cognition. Washington, DC, US: American Psychological Association. ISBN 1-4338-1204-5 (Hardcover); 978-1-4338-1204-0 (Hardcover).

Samuel J. Gershman. 2018. The Successor Representation: Its Computational Logic and Neural Substrates. *The Journal of Neuroscience*, 38(33): 7193.

Staiger, J. F.; and Petersen, C. C. H. 2021. Neuronal Circuits in Barrel Cortex for Whisker Sensory Perception. *Physiological Reviews*, 101(1): 353–415. Publisher: American Physiological Society.

Stolcke, A.; and Omohundro, S. M. 1994. Best-first Model Merging for Hidden Markov Model Induction. ArXiv:cmp-lg/9405017.

Tang, M.; Barron, H.; and Bogacz, R. 2023. Sequential Memory with Temporal Predictive Coding. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems*, volume 36, 44341–44355. Curran Associates, Inc.

Tolman, E. C. 1948. Cognitive maps in rats and men. *Psychological Review*, 55(4): 189–208.

Whittington, J. C. R.; McCaffary, D.; Bakermans, J. J. W.; and Behrens, T. E. J. 2022. How to build a cognitive map. *Nature Neuroscience*, 25(10): 1257–1272.

Whittington, J. C. R.; Muller, T. H.; Mark, S.; Chen, G.; Barry, C.; Burgess, N.; and Behrens, T. E. J. 2020. The Tolman-Eichenbaum Machine: Unifying Space and Relational Memory through Generalization in the Hippocampal Formation. *Cell*, 183(5): 1249–1263.e23.