# LEARNING TRANSFERABLE TASK SCHEMAS BY REPRESENTING CAUSAL INVARIANCES

**Tamas J. Madarasz**
University of Oxford
tamasmadarasz1@gmail.com

**Timothy E. Behrens**
University of Oxford
behrens@fmrib.ox.ac.uk

## ABSTRACT

Real-world tasks are often compositional and possess causal structure over component subtasks that is independent of a particular environment and the corresponding low-level observations. While rewards provide a natural way of controlling the behaviour of an agent solving such tasks, under the standard Markov assumption immediate rewards depend only on the current state and action. In contrast, if rewards have complex causal dependencies on the agent's history (such as collection of specific rewards or completion of previous subtasks), the Markov assumption either forces the state representation to encompass the whole history of the agent, or necessitates finding an effective way of compressing this history into a belief state on which to act. It is not clear how to achieve this while also allowing better generalization to novel environments, where one intuition is that using disentangled causal factors to capture the underlying generative process of a task is the beneficial approach. To address this problem, we propose a memory based meta-learning architecture that allows transfer of task structure through a dual approach: by basing predictions about rewards (including subtask completions) on an external memory indexed by a learned task decomposition, while also learning to infer and represent permutation invariances between different subtasks and rewards. This representation of causal and temporal dependencies allows the transfer of the task structure to novel environments, while also enabling the algorithm to trade off visiting currently rewarding states against long-term consequences.

## 1 INTRODUCTION

What could be more pleasant than inviting over one's friends for a well-organized dinner party to a new home? While certainly desirable, such feats of human organization and knowledge transfer are hard to capture algorithmically, as they rely on complex structural knowledge about the task at hand, allowing one to accomplish it more-or-less successfully even after relocating to a new flat, country, or continent. In our particular example, organising a dinner party relies, among other things, on the knowledge that food should (mostly) be prepared before guests arrive, and that shopping needs to be done before cooking. But also that shopping for ingredients itself is a (mostly) order-independent process, allowing us to plan the most efficient route within and between stores and their aisles, whereas cooking a recipe (mostly) requires following steps in strict order. More generally,machines autonomously performing sets of tasks will need to be able to reason about the temporal and causal dependencies and invariances involved to be able to solve these efficiently across varied environments.

In this paper we explore a memory-based meta-learning approach to learn to represent different phases, or 'task states' of compositional tasks, and the relationships between these task states, in order to successfully complete tasks with shared structure across different settings. In particular, we use a memory-augmented neural network (Santoro et al., 2016) as a generative model to represent task states as well as order-dependency, predict eligible subtasks and rewards, and plan ahead to complete subtasks in the order promising of the highest total return.

## 2 BACKGROUND

Markov Decision Processes (MDPs) have been successful as a model for solving many simulated and real-world problems while using reward functions obeying the Markov property, where the reward distribution is a function of the current state, action and arrival state only. It has long been observed, however, that natural rewards guiding human and animal behaviour, or determining desirable performance criteria in robotics can often be non-Markovian (Bacchus et al., 1996; Littman et al., 2017) when specified using intuitive, tractable state spaces. In the case of Non-Markovian Reward Decision Processes (NM-RDPS), one approach to this problem is to use the entire history of the agent as an extended state space on which to define the reward distribution, however this is clearly problematic because of the resulting combinatorial explosion. Earlier work has thus focused on learning a minimal extension of the state space that makes rewards Markovian, using the formulations of linear temporal logic, or finite state automata (Li et al., 2016; Camacho et al., 2017; Brafman et al., 2018; Icarte et al., 2018). While the motivation behind most of such work is to reduce the size of the extended state space to mitigate its effect on the sample complexity of learning a particular task, here we focus expressly on generalization, namely, learning the *structure* of the reward function for a family of tasks. In the next sections we outline our approach, its relationship to learning state space models and causality, and describe our implementation as a factorized recurrent neural network.

## 3 MOTIVATION

### 3.1 SEMI-MARKOV TASK STATES

The core of our approach is to augment the state space of a task specified by a non-Markov reward function (NMRF) using an explicit task state variable. This allows us to segment the task into subparts in a generalizable way, such that future rewards are independent of the history during *previous* task states, when conditioning on the current task state. This idea is related to so-called semi-Markov options (Sutton et al., 1999), where the termination condition is a function only of the history during the execution of the option. Hence we refer to this type of segmentation as semi-Markov task states. Unlike in the case of options, however, the task states are inherent in the structure of the task itself, and not simply a way to define temporal abstraction over actions. We make the assumption that the true reward function at a specific time is always expressible as a function of the task identity $T_t$ and the complete state and reward history during the episode $\mathcal{H} = (s_0, s_1, r_1, ...st, r_t)$ history, where we leave out the history of actions for convenience and condition rewards on the arrival state, though these assumptions are not central to our approach. Given this assumption, we desire the decomposition of the task into task states to be generalizable and minimal, in the sense that we describe below.

Intuitively, to learn a general task *schema*, we want the transition between task states to depend as much as possible on general features (such as collecting a certain number of rewards signalling the completion of some subtask), and as little as possible on features specific only to the current environment, such as sensory features attached to the rewards, or the states visited to acquire them. In our original example, if we learned that shopping is an order-independent task state of collecting $n$ items on the shopping list, we know to transition to the next task state after $n$ reward signals, independently of the order in which we picked up the items, or the specific sensory stream that accompanied the experience. At the same time, the local history during the 'shopping' task state is sufficient for guiding the collection of rewards by keeping track of the items already bought and providing information about where the remaining items might be. This way we are not forced into an unnecessary ordering of collecting rewards/completing subtasks that is present in some previous work (e.g.(Andreas et al., 2017)), unless this ordering is actually important for solving the overall task. The resulting invariance enables generalisability, since we can learn structure that is independent of specific features, as well as a policy that can exploit the invariance to plan more efficient trajectories in the state space, and should therefore be encouraged as part of the objective in the multi-task learning setting. In contrast, if the lower-level details of a subtask (as represented by reward features or state visitations) are important for the evolution of the reward function, these should also factor into task state transitions, creating an order-dependent evolution of task states. This, however, might result in having to evaluate the effect of an exponentially growing number of different orderings on the evolution of the

reward function when planning, and limited generalisability for a different setting of sensory/reward features in a new instantiation of the task.

### 3.2 ORDER-DEPENDENCE AND INVARIANT PREDICTION

This objective also corresponds to identifying variables that provide *invariant predictions* across environments as suggested in (Peters et al., 2015). The notion of environments in this case encompasses both different task instantiations, as well as the different interventions by the agent through its actions and collecting rewards. As the agent interacts with the same task instantiation across several episodes, and stores these interactions in memory, this information, together with knowledge of the task structure, allows the agent the ability of counterfactual reasoning about task-policy combinations it never encountered. In particular, the permutation invariance inherent in order-independent subtasks, and found in the overall task structure can allow the agent to predict non-Markovian rewards without previous experience of the same evolution of the reward function. Further, it can lead to identifying the corresponding orderings as non-causal, and therefore unnecessary for reward-prediction in the future. In other words, the order in which rewards are collected in an order-independent subtask can be set arbitrarily, without influencing downstream rewards, and therefore does not lead to invariant predictions in the sense of a potential causal factor. For a detailed discussion relating invariant predictions and causality, we refer the reader to (Peters et al., 2015).

We use these insights into the desirable inductive biases and representations to construct a model that learns task states predicting rewards in an NM-RDP, while encouraging generalizability by regularizing the flow of specific information into the task state transition process.

## 4 ALGORITHM AND IMPLEMENTATION

We implement the learning of task states in a neural generative model using a recurrent implementation of the variational auto-encoder (VAE) framework (Kingma & Welling, 2013; Rezende et al., 2014). The model learns a factorized state space model (Dynamic Bayesian Network) that handles an external memory, similarly to previous approaches implementing a generative temporal model with external memory to make sensory predictions (Gemici et al., 2017; Fraccaro et al., 2018), predict returns for better RL (Wayne et al., 2018), or generalize structural knowledge (Whittington et al., 2018).

### 4.1 GENERATIVE MODEL

The agent's generative model (Fig. 1a,b) predicts rewards in the arrival state and can be used to construct reward maps for the environment to select actions. The generative model factorizes as

$$p_\theta(\mathbf{r}_{\leq T}, \mathbf{c}_{\leq T}, \mathbf{g}_{\leq T} | \mathbf{s}_{\leq T}) = \prod_{t=1}^{t=T} p_\theta(g_t | g_{t-1}, r_{t-1}) p_\theta(c_t | c_{t-1}, g_t, K_t, r_{t-1}) p_\theta(r_t | c_t, M_t, s_t),$$

where $c$ is the task state (implemented as a vector) and $M$ a memory of transitions (including previous episodes) indexed by task state. The scalar $g$ is a gating variable that gates the effect of state information on the task state c, with this state information being represented by a compressed state-reward history K (implemented as the hidden unit of an LSTM with parameters $\theta_K$). Thus the task state moves forward given the reward in the current step, and optionally the identity of the reward as signalled by the state information, as would be desirable in order-dependent parts of a task where differences in histories can lead to very different future predictions. All of the transition densities were modelled as Gaussians with diagonal covariances, with the means and covariances computed by MLPs. Both $K$ and $M$ are deterministic, in the sense that new transitions are added in a deterministic manner without any sampling. The predictive densities for rewards come from using the task state variable $c_t$ and arrival state $s_{t+1}$ as keys to the memory $M_t$ to retrieve sufficient statistics from the current and previous episodes, which are then passed into a small MLP (the *reward network* with parameters $\theta_r$) to give the parameters for a scalar Gaussian density. In line with our motivation above, the current task state $c_t$ at time $t$ in episode $e_t$ retrieves memories from the current episode (working or short-term memory) and previous episodes (long-term memory) using cosine similarity attention, to condition predictions using the part of history under similar task states, in

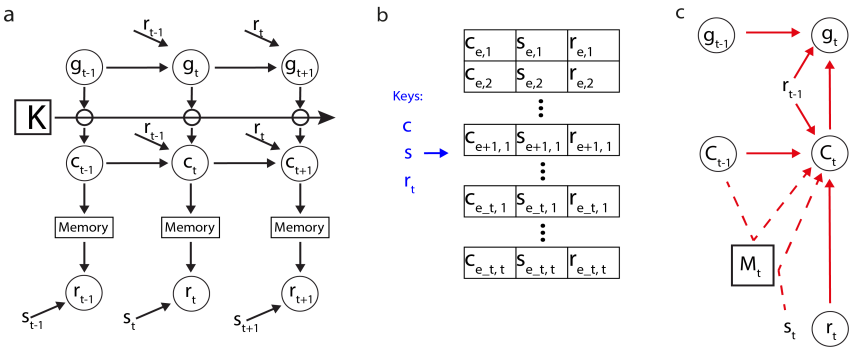


Figure 1: a, Generative model of rewards $r$ using task states $c$ and memory $M$. Circled letters denote stochastic variables, whereas boxed letters are deterministic. While we show arrows from the state $s$ to the reward $r$, this influence also goes through the memory, playing the part of attention keys. $K$ is a deterministic recurrent neural network summarizing state-reward information, and $g$-s are ariables gating the flow of this information into the task states $c$. b, External memory storing transitions from current and previous episodes. c, Inference network. Dashed arrows indicate optional computations helping the inference e.g. by retrieving values of past task states $c$ from the memory using $r$ and $s$ as keys.

a soft way. We used a set of three sufficient statistics as input to the reward network. First, from previous episodes, the weighted sum of rewards retrieved using the keys $c_t$ and $s_t$ per episode, averaged over episodes, $\frac{1}{e_t-1}\sum_{e=1}^{e=e_t-1}\sum_k sim(c_t, c_{e,k}) \cdot sim(s_t, s_{e,k}) \cdot r_{e,k}$. Here $sim$ denotes cosine similarity, and the first and second subscripts are episode number and timestep respectively. Second, the maximum reward retrieved with the keys of $c_t$ and $s_t$ across all previous episodes in the memory $\max_{e<e_t,k} sim(c_t, c_{e,k}) \cdot sim(s_t, s_{e,k}) \cdot r_{e,k}$, where $e_t$ is the current episode. And third, the sum of rewards already collected during the current episode $\sum_k sim(c_t, c_{e_t,k}) \cdot sim(s_t, s_{e_t,k}) \cdot r_{e,k}$. This gives us the flexibility to predict locally non-Markovian rewards, e.g. rewards that can be 'picked up' once, or that deplete according to some well-defined rule, or indeed are Markovian.

### 4.2 Inference model and two-filter smoothing

We approximate the intractable filtering and smoothing posteriors, $q_\phi(\mathbf{g}_{\leq t}, \mathbf{c}_{\leq t}|\mathbf{r}_{\leq \mathbf{t}}, \mathbf{s}_{\leq \mathbf{t}})$ and $q^s_\phi(c_t|\mathbf{r}_{\leq \mathbf{T}}, \mathbf{s}_{\leq \mathbf{T}})$ respectively using variational inference with auto-regressive posteriors. Smoothing is necessary in our model since later rewards are causally dependent on earlier ones, therefore a fully informative inference over task states and order dependence is only possible once we observe future consequences. While at the action selection step we can only use the prior distribution over the task state as computed from the previous, filtered posterior $p_\theta(c_t|c^{filtered}_{t-1}, r_{t-1}, g_t, K_t)$ we use the smoothing posterior to overwrite the corresponding filtered ones in the memory $M$ at the end of each episode. This way, we are able to make better inferences over the task states and plan more optimally on later episodes. In our implementation with recurrent neural networks, it will be important to be able to compute a per-step variational lower bound (ELBO), so we use a variational implementation of a two-filter smoother (Briers et al., 2004), where the smoothing posterior at time $t$ is computed using the product of two independent (a forward and backward) filters. The form of the variational filtering and smoothing posteriors is given in the Supplementary Information.

The VAE framework then allows us to represent the component conditional densities by neural networks, sample the stochastic variables, and jointly optimize the generative, posterior, smoothing, LSTM (for computing $K$), and reward network parameters. We maximize two separate ELBOs, the filtering ELBO at every step of the agent wrt. parameters $\theta, \theta_r, \theta_K$ and $\phi$, and a smoothing ELBO at the end of the episode when we run the backward information filter to the beginning of the episode, and optimize wrt. parameters $\theta, \theta_K$ and $\phi_2$.

### 4.3 REWARD MAPS, CONTROL, AND PLANNING

To test the effectiveness of this framework, we use a heuristic strategy for control and planning. Rather than applying Q-learning anew for each task and task state, we will compute an approximation based on the successor representation (Dayan, 1993). If, for example, rewards are determined by the arrival state $r \equiv r(s')$, this framework allows the computation of the value function as the product of expected state occupancies, and a linear weight vector $\mathbf{w}$ obeying $r(s') = \Phi(s') \cdot \mathbf{w}$, where $\Phi(s')$ is the state embedding of $s'$. Though the non-Markovian nature of the reward function violates the assumptions under which this factorization normally holds, this estimate still provides a suitable heuristic for control that quickly adapts to new reward locations, without attempting to exactly solve e.g. the travelling salesman problems that can arise in our problem setting.

To instantiate a successor representation based approach, our generative model can predict a reward $r^p(s)$ given a hypothetical visit to any state $s$, given a current task state $c_t$. We can therefore approximate, by cycling through previously visited states as a form of replay, a conditional linear reward map $\mathbf{w}_t$, such that the reward expected for visiting state $s'$ during the current task state $c_t$ and with memory $M_t$ is $r^p(s') = \Phi(s') \cdot \mathbf{w_t}$.

We can further define a value function for the task state c, $V(c, s, h^c)$, where $h^c$ is the history during the task state $c$. When learning this value function in practice (using an MLP), we replace the history $h^c$ with the estimate of collected rewards during task state $c$, retrieved from the memory: $\sum_k (c_t \cdot c_{e_t,k}) \cdot r_{e,k}$. Our forward dynamics model over the task states also allows for planning in this abstract space (Oh et al., 2017; Silver et al., 2017), by imagining the consequences of visiting a state $s'$, receiving reward $r$ and transitioning to task state $c'$. Importantly, the value of the gating variable $g$ determines the extent to which planning over task states is necessary: in case of order-independence, the consequences of different orderings on future task states do not need to be compared. We use the approximate relationship $V(c, s, h^c) \approx r^p(s') + V(c', s', \emptyset)$, and add the product of $g$ and the imagined value, $g * V(c', s', \emptyset)$ to the predicted immediate reward $r^p(s')$ when learning the reward map $\mathbf{w_t}$ through 'replay'.

## 5 EXPERIMENTS

We ran simulations on 8 by 8 (5 by 5 for Task 3) tabular grid-world environments under three conditions (Fig. A1). In Task 1, the *sequential task*, three identical reward items (each worth 5 points) appeared sequentially in predetermined order. Each item (except the first) only appeared once the previous one was collected, and was removed by the agent stepping onto the corresponding grid in the environment and collecting the corresponding number of points. The location of all three rewards changed every 30 episodes. Reward locations were not signalled to the agent, but task changes were signalled, and the memory wiped at those points, so there was no interference from previous tasks. In the order-independent task, Task 2, three rewards (each worth 5 points) at distinct locations were concurrently available at the beginning of the task, and a fourth (worth 10 points) appeared once all of these were collected and removed, in any order. Finally, in the order-dependent task, Task 3, the last reward only appeared if the first three were collected in a particular, randomly determined order. For all tasks, returns were calculated using a discount factor of $\gamma = 0.99$, and episode-lengths were capped at 100. Figure 2 shows the results compared to controls using double Q-learning and the A3C algorithm, with the number of runs $n = 5$ in all cases. Further, we also found that as expected, the absolute value of g was indeed on average much larger during task 3 and smallest in task 2.

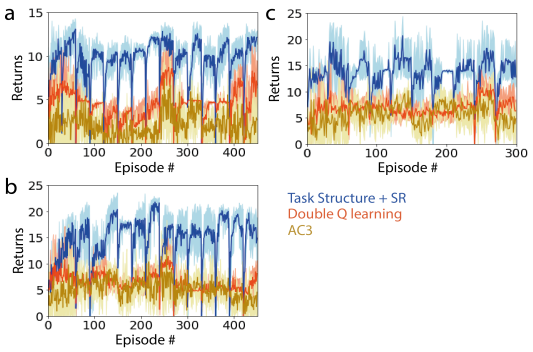


Figure 2: Results of the experiments: a, Sequential task. b, Order-independent two-stage task. c, Order-dependent two stage task.

## 6 RELATED WORK AND CONLCUSION

We introduced a framework to decompose a task depending on its temporal subtask depen-

dencies in a generalisable way using a neural state space model. Unlike previous works we don't assume predefined policies for subtasks (Sohn et al., 2020), a set propositional symbols corresponding to important events (Icarte et al., 2018), or entrenched orderings (Andreas et al., 2017) and in ongoing work we are focusing on the implementation of more sophisticated control approaches. Finally, other recent approaches focusing on restricting information flow for generalizable hierarchical reinforcement learning (Goyal et al., 2020) also offer further potential for a fruitful synthesis of ideas.

## REFERENCES

Jacob Andreas, Dan Klein, and Sergey Levine. Modular multitask reinforcement learning with policy sketches. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Fahiem Bacchus, Craig Boutilier, and Adam Grove. Rewarding behaviors. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*. AAAI Press, 1996. ISBN 026251091X.

Ronen Brafman, Giuseppe De Giacomo, and Fabio Patrizi. Ltlf/ldlf non-markovian rewards. In *AAAI Conference on Artificial Intelligence*, 2018.

Mark Briers, Arnaud Doucet, and Simon Maskell. Smoothing algorithms for state-space models. In *in Submission IEEE Transactions on Signal Processing*, 2004.

Alberto Camacho, Oscar Chen, Scott Sanner, and Sheila A. McIlraith. Non-markovian rewards expressed in ltl: Guiding search via reward shaping. In *Proceedings of the Tenth International Symposium on Combinatorial Search, SOCS 2017, 16-17 June 2017, Pittsburgh, Pennsylvania, USA*, 2017.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Comput.*, 1993. ISSN 0899-7667.

Marco Fraccaro, Danilo Jimenez Rezende, Zwols Yori, Alexander Pritzel, S. M.Ali Eslami, and Viola Fabio. Generative temporal models with spatial memory for partially observed environments. In *Proceedings of 35th International Conference on Machine Learning, ICML 2018*, 35th International Conference on Machine Learning, ICML 2018, 2018.

Mevlana Gemici, Chia-Chun Hung, Adam Santoro, Greg Wayne, Shakir Mohamed, Danilo J. Rezende, David Amos, and Timothy Lillicrap. Generative temporal models with memory. *arXiv preprint 1702.04649*, 2017.

Anirudh Goyal, Shagun Sodhani, Jonathan Binas, Xue Bin Peng, Sergey Levine, and Yoshua Bengio. Reinforcement learning with competitive ensembles of information-constrained primitives. In *International Conference on Learning Representations*, 2020.

Rodrigo Toro Icarte, Toryn Klassen, Richard Valenzano, and Sheila McIlraith. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint 1312.6114*, 2013.

Xiao Li, Cristian-Ioan Vasile, and Calin Belta. Reinforcement learning with temporal logic rewards, 2016.

Michael L. Littman, Ufuk Topcu, Jie Fu, Charles Isbell, Min Wen, and James MacGlashan. Environment-independent task specifications via gltl. *arXiv preprint 1704.04341*, 2017.

Junhyuk Oh, Satinder Singh, and Honglak Lee. Value prediction network. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7192-value-prediction-network.pdf.

Jonas Peters, Peter Bhlmann, and Nicolai Meinshausen. Causal inference using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78, 01 2015. doi: 10.1111/rssb.12167.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, pp. II1278II1286. JMLR.org, 2014.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, New York, New York, USA, 2016.

David Silver, Hado van Hasselt, Matteo Hessel, Tom Schaul, Arthur Guez, Tim Harley, Gabriel Dulac-Arnold, David Reichert, Neil Rabinowitz, Andre Barreto, and Thomas Degris. The predictron: End-to-end learning and planning. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, International Convention Centre, Sydney, Australia, 2017. PMLR.

Sungryull Sohn, Hyunjae Woo, Jongwook Choi, and Honglak Lee. Meta reinforcement learning with autonomous inference of subtask dependencies. 2020.

R.S. Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112:181–211, 1999.

Greg Wayne, Chia-Chun Hung, David Amos, Mehdi Mirza, Arun Ahuja, Agnieszka Grabska-Barwinska, Jack W. Rae, Piotr Mirowski, Joel Z. Leibo, Adam Santoro, Mevlana Gemici, Malcolm Reynolds, Tim Harley, Josh Abramson, Shakir Mohamed, Danilo Jimenez Rezende, David Saxton, Adam Cain, Chloe Hillier, David Silver, Koray Kavukcuoglu, Matthew Botvinick, Demis Hassabis, and Timothy P. Lillicrap. Unsupervised predictive memory in a goal-directed agent. *CoRR*, abs/1803.10760, 2018. URL http://arxiv.org/abs/1803.10760.

James C.R. Whittington, Timothy H. Muller, Shirley Mark, Caswell Barry, and Timothy E.J. Behrens. Generalisation of structural knowledge in the hippocampal-entorhinal system. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018.

# A SUPPLEMENTARY INFORMATION

## A.1 POSTERIOR DISTRIBUTIONS AND FORM OF THE VARIATIONAL BOUNDS

The variational filtering posterior is given by

$$q_\phi(\mathbf{g}_{\leq t}, \mathbf{c}_{\leq t}|\mathbf{r}_{\leq \mathbf{t}}, \mathbf{s}_{\leq \mathbf{t}}) = \prod_{\tau=1}^{\tau=t} q_\phi(g_\tau|g_{\tau-1}, c_\tau, r_{t-1}, r_t) \cdot q_\phi(c_\tau|c_{\tau-1}, g_{\tau-1}, M_t, s_t, K_t, r_{t-1}, r_t),$$

whereas the smoothing posterior for $c$ is given by

$$q_{\phi,\phi_2}(c_t|\mathbf{r}_{\leq \mathbf{T}}, \mathbf{s}_{\leq \mathbf{T}}) \propto q_\phi(c_t|\mathbf{r}_{\leq \mathbf{t}}, \mathbf{s}_{\leq \mathbf{t}}) \cdot \int q_{\phi_2}(c_t|\tilde{c}_{t+1}, M_t, r_t, r_{t+1}) \cdot \prod_{\tau=T}^{\tau=t+2} q_{\phi_2}(\tilde{c}_{\tau-1}|\tilde{c}_\tau, M_{\tau-1}, r_{\tau-1}, r_\tau) \cdot q_{\phi_2}(\tilde{c}_T|r_T) d\tilde{\mathbf{c}},$$

where $q_{\phi_2}(\tilde{c}_\tau)$ are the (independently) computed *information filter* posteriors using backward filtering.

From Jensen's inequality we have:

$$\log p_\theta(\mathbf{r}_{\leq T}|\mathbf{s}_{\leq T}) \geq \mathbb{E}_{q_\phi(\mathbf{g}, \mathbf{c}_{\leq T}|\mathbf{r}, \mathbf{s}_{\leq T})} \frac{\log p_\theta(\mathbf{g}, \mathbf{c}, \mathbf{r}_{\leq T}|\mathbf{s}_{\leq T})}{\log q_\phi(\mathbf{g}, \mathbf{c}_{\leq T}|\mathbf{r}, \mathbf{s}_{\leq T})},$$

and the filtering variational lower bound is given by

$$\mathcal{F} = \sum_{t=1}^{T} \mathbb{E}_{\prod_{\tau=1}^{t} q_\phi(\mathbf{g}, \mathbf{c}_{\leq \tau}|\mathbf{r}, \mathbf{s}_{\leq \tau}, \mathbf{g}, \mathbf{c}_{<\tau})} \Big[ \log p_\theta(g_t|g_{t-1}, r_{t-1}) + \log p_\theta(c_t|c_{t-1}, g_t, K_t, r_{t-1}) + \log p_\theta(r_t|c_t, M_t, s_t) - \log q_\phi(g_\tau|g_{\tau-1}, c_\tau, r_{t-1}, r_t) - \log q_\phi(c_\tau|c_{\tau-1}, g_{\tau-1}, k_t, r_{t-1}, r_t) \Big],$$

where the dependence on $s_t$ is always realized through a retrieval of memories from $M_t$, rather than directly feeding the state into the generative/recognition networks. We maximize this ELBO minus a regularization penalty term for the norm of $g$.

The smoothing ELBO is given by

$$\mathcal{F} = \sum_{t=1}^{T} \mathbb{E}_{\prod_{\tau=1}^{t} q^s_{\phi,\phi_2}(\mathbf{g}, \mathbf{c}_{\leq \tau}|\mathbf{r}, \mathbf{s}_{\leq T}, \mathbf{g}, \mathbf{c}_{<\tau})} \Big[ \log p_\theta(g_t|g_{t-1}, r_{t-1}) + \log p_\theta(c_t|c_{t-1}, g_t, K_t, r_{t-1}) + \log p_\theta(r_t|c_t, M_t, s_t) - \log q^s_{\phi,\phi_2}(g_t|g_{t-1}, c_t, r_{t-1}, r_t) - \log q^s_{\phi,\phi_2}(c_t|c_{t-1}, g_{t-1}, r_{\leq T}, s_{\leq T}, M_T) \Big].$$

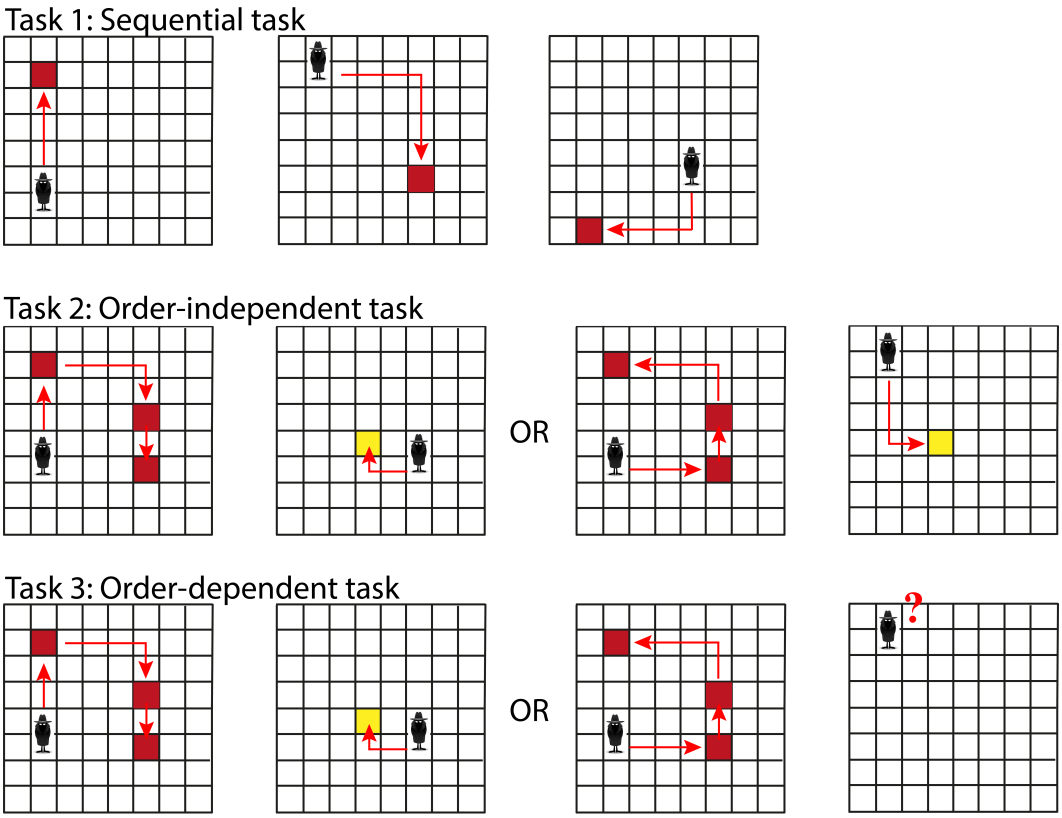


Figure A1: Illustration of the tasks used for our experiments