Deep reinforcement learning with time-scale invariant memory

Md Rysul Kabir, James Mochizuki-Freeman, Zoran Tiganj

Department of Computer Science
Luddy School of Informatics, Computing, and Engineering
Indiana University Bloomington
mdrkabir@iu.edu, jmochizu@iu.edu, ztiganj@iu.edu

Abstract

The ability to estimate temporal relationships is critical for both animals and artificial agents. Cognitive science and neuroscience provide remarkable insights into behavioral and neural aspects of temporal credit assignment. In particular, scale invariance of learning dynamics, observed in behavior and supported by neural data, is one of the key principles that governs animal perception: proportional rescaling of temporal relationships does not alter the overall learning efficiency. Here we integrate a computational neuroscience model of scale invariant memory into deep reinforcement learning (RL) agents. We first provide a theoretical analysis and then demonstrate through experiments that such agents can learn robustly across a wide range of temporal scales, unlike agents built with commonly used recurrent memory architectures such as LSTM. This result illustrates that incorporating computational principles from neuroscience and cognitive science into deep neural networks can enhance adaptability to complex temporal dynamics, mirroring some of the core properties of human learning.

Introduction

Learning temporal relationships between cause and effect is critical for successfully obtaining rewards and avoiding punishments in a natural environment. Humans and many other animals can estimate the temporal duration of events and use that estimate as an integral component of decision-making. Furthermore, the ability to do this rapidly and flexibly across a wide range of temporal scales has been critical for survival. While machine learning systems also possess the capacity to represent the elapsed time, typically via recurrent connections, they often struggle with learning temporal relationships, especially when those are extended over multiple scales.

The mammalian ability to estimate time and learn exhibits scale invariance across a wide range of temporal scales, spanning from seconds to several minutes (Buhusi and Meck 2005; Gibbon 1977; Buhusi et al. 2009; Balci and Freestone 2020; Gallistel and Gibbon 2000; Balsam and Gallistel 2009; Gallistel and Shahan 2024). A scale invariant system has a linear relationship between the mean estimated time and the actual time, with a constant coefficient of variation. In other words, the relative uncertainty or error in time estimation remains constant as the duration increases. This is known as the Weber-Fechner law (Fechner 1860/1912), which states that the just noticeable difference between two stimuli is proportional to the magnitude of the stimuli. Hence the ratio between these two quantities is constant, implying that the perceived magnitude is on a logarithmic scale. This law is foundational for understanding mammalian perceptions and spans virtually all perceptual domains except angles (Gibbon 1977; Wilkes 2015). Scale invariance in learning is demonstrated in classical conditioning where animals observe a salient stimulus followed by a reward. When the temporal distance between stimulus and reward is rescaled by the same factor as between two stimuli, the number of trials the animal needs to learn the value of the stimulus remains unchanged. While previous studies have demonstrated this effect on the order of about a minute (Gallistel and Gibbon 2000; Balsam and Gallistel 2009), recent work has demonstrated that it holds for at least 16 minutes (Gallistel and Shahan 2024). Contrary to biological organisms, the performance of machine learning systems is typically not scale invariant. Machine learning systems tend to perform well only at a limited set of scales and require adjustments of hyperparameters such as learning rate, temporal resolution and temporal discounting to learn problems at different scales.

A number of neuroscience studies have investigated the neural underpinning of temporal representations in tasks such as interval timing and temporal bisection (Emmons et al. 2017; Matell and Meck 2004; Kim et al. 2013, 2017; Parker et al. 2014; Mello, Soares, and Paton 2015; Narayanan 2016; Gouvêa et al. 2015; Tiganj et al. 2017; Donnelly et al. 2015). Neural activity observed in several brain regions, including the hippocampus, prefrontal cortex and striatum, was often characterized by one of two traits: (1) ramping/decaying activity, where neurons monotonically increase/decrease their firing rate as a function of time following some salient stimulus (such as the onset of the interval timing cue); and (2) sequential activity, where neurons activate sequentially following some salient stimulus, with each neuron elevating its firing rate for a distinct period of time. These sequentially activated neurons, also known as *time cells* (as a temporal analog of *place cells* (O'Keefe 1976)) have been observed in the hippocampus

and the prefrontal cortex in several studies over the recent years (Pastalkova et al. 2008; MacDonald et al. 2011; Salz et al. 2016; Cruzado et al. 2020; Eichenbaum 2014; MacDonald and Tonegawa 2021). Sequential neural activity is also scale invariant: the width of the temporal windows in time cells increases with the peak time and the ratio of the peak times of the adjacent cells is constant (geometric progression), indicating uniform spacing along a logarithmic axis (Cao et al. 2022).

In this study, we draw on insights from neuroscience and cognitive science to enable more flexible learning in deep reinforcement learning (RL) agents by incorporating scale invariant temporal memory. The memory model is based on previous work in computational neuroscience (Shankar and Howard 2012, 2013), which has been used to explain a broad range of phenomena in neuroscience (Howard et al. 2014), including the emergence of time and place cells, as well as findings from cognitive psychology memory experiments, such as free recall and judgment of recency (Howard et al. 2015; Tiganj, Tang, and Howard 2021). We integrate scale invariant memory into RL agents and evaluate their performance across tasks designed to span a wide range of temporal scales. We compare the performance and neural activity of proposed agents to agents built with standard memory architectures, such as LSTM (Hochreiter and Schmidhuber 1997) and simple RNN. Through theoretical analysis and experiments, we demonstrate that agents with scale invariant memory effectively generalize and maintain strong performance across rescaled temporal relationships.

Prior work

Previous computational work proposed that ramping/decaying activity and time cells provide a representation essential for timing and learning temporal relationships (Balci and Simen 2016; Howard et al. 2014). This has also been studied in the RL context (Petter, Gershman, and Meck 2018; Namboodiri 2022; Ludvig, Sutton, and Kehoe 2008). A number of computational neuroscience studies have closely examined and attempted to model neural activity during interval timing and similar time-related tasks (Grossberg and Schmajuk 1989; Wang et al. 2018; Jazayeri and Shadlen 2015; Cueva et al. 2020; Pérez and Merchant 2018; Raphan, Dorokhin, and Delamater 2019). Together, these models provide remarkable insights into neural mechanisms of temporal learning from computational neuroscience and cognitive science perspectives. Our efforts focus on examining temporal learning in artificial RL agents trained using error backpropagation. We believe this effort complements computational neuroscience efforts for building neural models of time-scale invariant learning since we specifically evaluate agents based on a computational neuroscience model of memory.

Deverett et al. (2019) studied interval reproduction in deep RL agents and provided valuable insights about the capabilities of these agents. Our work extends this approach, as we closely examine the neural activity of the artificial agents and conduct experiments at different temporal scales. Previous work has also shown that under particular circumstances (mnemonic demand), neural activity in deep RL agents resembles the activity of time cells (Lin and Richards 2021). Other types of brain-like neural activity have also been observed in deep learning systems (Sorscher et al. 2019; Schaeffer, Khona, and Fiete 2022), which in some cases led to improved performance of deep RL agents (Banino et al. 2018). Furthermore, recent computational models of hippocampal activity have been linked to modern deep learning architectures, such as transformers (Whittington, Warren, and Behrens 2021). Our work complements these efforts in brain-inspired AI by comparing neural activity in biological and artificial systems and evaluating whether systems whose activity is closer to biological counterparts indeed perform better.

Neural networks that use temporal convolution to construct an offline version of scale invariant memory have been proposed in Jacques et al. (2021, 2022), but these have not been explored in the context of online temporal learning or deep RL. An approach similar to ours has been used to develop systems with scale invariant temporal discounting (Tano, Dayan, and Pouget 2020; Masset et al. 2023; Momennejad and Howard 2018; Tiganj et al. 2019). A similar approach has also been used to build systems that can learn to represent variables such as numerosity and position (Maini et al. 2023; Mochizuki-Freeman, Kabir, and Tiganj 2024). These approaches are complementary to ours since we focus on scale invariance in temporal memory of deep RL agents.

Environments

Interval timing

The interval timing environment was inspired by a neuroscience study where rats had to discriminate between two groups of intervals, long and short, each having three possible durations (Kim et al. 2013) (Fig. 1A). The interval begins when the agent crosses the start line (indicated by the red line in Fig. 1B). The duration of the interval is randomly chosen from six equally probable durations. After the end of the interval period, the central bridge of the T-maze drops, and the agent freely navigates towards the end of the track. When the agent reaches the end, it has to choose one of the goal locations (indicated by the blue lines in Fig. 1B) depending on whether the interval was from the long or short group of intervals. We used two versions of this environment: one with 3D realistic visual observations developed using the open-source PyBullet physics engine (Coumans and Bai 2021), and the other that consists of a 1D observation space. The purpose of the simple 1D environment was to facilitate interpretability of the results, while the 3D realistic environment was used to evaluate the scalability of the proposed approach.

In the 3D environment, at each time step, the agent receives a three-dimensional pixel observation of shape $60 \times 60 \times 3$ and performs any of the three possible actions: *left*, *right* and *forward*. The *forward* action moves the agent straight towards the end of the track at a fixed speed of 1cm/step. Each *left* and *right* action realigns the agent in 7.5° increments to a max/min of $\pm 15^\circ$ along the straight

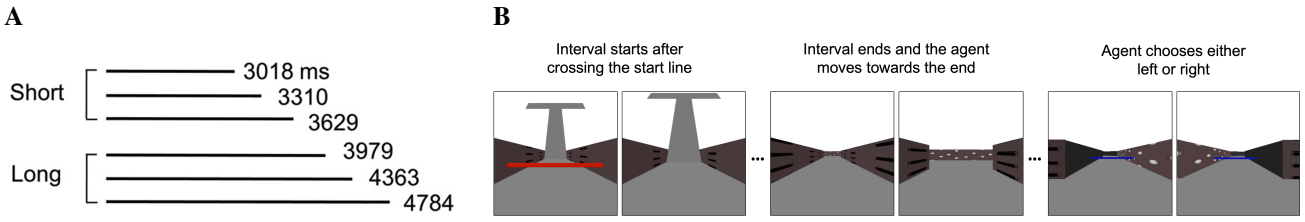

Figure 1: **A.** The six temporal intervals used in the task. At each trial, a random interval is selected and the agent has to indicate whether the interval was long or short. **B.** Schematic of the environment. After the agent crosses the start line, one of six delay intervals is presented.

portion of the T-maze and without a bound at the end. Once the agent reaches the goal positions, it receives a reward of either 10 for a correct or 0 for an incorrect choice. It also receives a reward of -0.1 if it tries to turn beyond $15°$ in either direction when on the straight track, or if it collides with the back wall of the environment.

In the simple version of the environment (top row in Fig. 3B), the observation space is one-dimensional such that a δ pulse is introduced after the fixation period to signal the beginning of an interval, followed by a second δ pulse that marks the end of the interval. The interval is followed by a delay period, after which the agent chooses either a "left" or "right" action, similar to the 3D environment. In this and subsequent environments, the agent receives a reward of 1.0 for a correct choice and -1.0 for an incorrect choice.

Interval discrimination

This simple environment was based on a duration discrimination task (Genovesio, Tsujimoto, and Wise 2009). In each trial, two stimuli are presented sequentially with a delay period separating them. The agent is required to determine which of the two stimuli has a longer duration. The observation space is one-dimensional. Consistent with the 1D interval timing task, the onset and termination of each interval are signaled by a δ pulse.

Delayed-match-to-sample

We used an existing delayed-match-to-sample environment from NeuroGym (Molano-Mazon et al. 2022). This environment was inspired by a common memory task (Miller, Erickson, and Desimone 1996) in which a sample stimulus is followed by a delay period and a test stimulus. To receive a reward, the participant needs to determine whether the test stimulus matches the sample stimulus. The observation space for this environment is a vector comprising three features.

Interval reproduction

This simple environment is inspired by the interval reproduction task described by Deverett et al. (2019). In each trial, two stimuli are presented sequentially, separated by an interval that the agent must later reproduce. During the reproduction phase, which occurs after the second stimulus, the agent must perform an action to replicate the interval within a 20% tolerance to receive a reward. In line with the other 1D tasks, the onset and termination of the interval are marked by a δ pulse.

Model

Our recurrent deep learning architecture consists of three major parts (Fig. 2): *encoder* (used only for the 3D interval timing environment), *core*, and *agent*. The *encoder* consists of three consecutive convolutional layers. The resulting output from the last convolutional layer is flattened and passed through a fully connected layer. This encoded representation is then fed into the *core*, the recurrent memory of our architecture. We used three different kinds of recurrent cores: RNN, LSTM, and the cognitively inspired RNN, described in detail below, which we abbreviate as CogRNN. Lastly, the output from the *core* is passed through two attention layers (for the 3D interval timing environment) or a dense layer (for other environments unless specified otherwise). This output is then fed into the agent part of the network, which has two branches: (i) policy network and (ii) value network. At every time step, the agent chooses an action depending on the output of the policy network. Outputs from the policy and value networks are used to calculate losses, which are then used in backpropagation for gradient-based parameter updates.

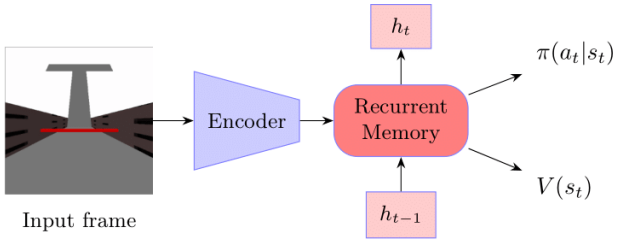

Figure 2: Architecture of the (RL) agent. Observations from the environment are processed by a convolutional neural network to extract feature representations. These features are then passed to a recurrent memory module (simple RNN, LSTM or CogRNN), which captures temporal dependencies and provides context for the policy network (π) and value network (V).

Scale invariant memory network

Building on models from computational and cognitive neuroscience (Shankar and Howard 2012; Howard et al. 2014),

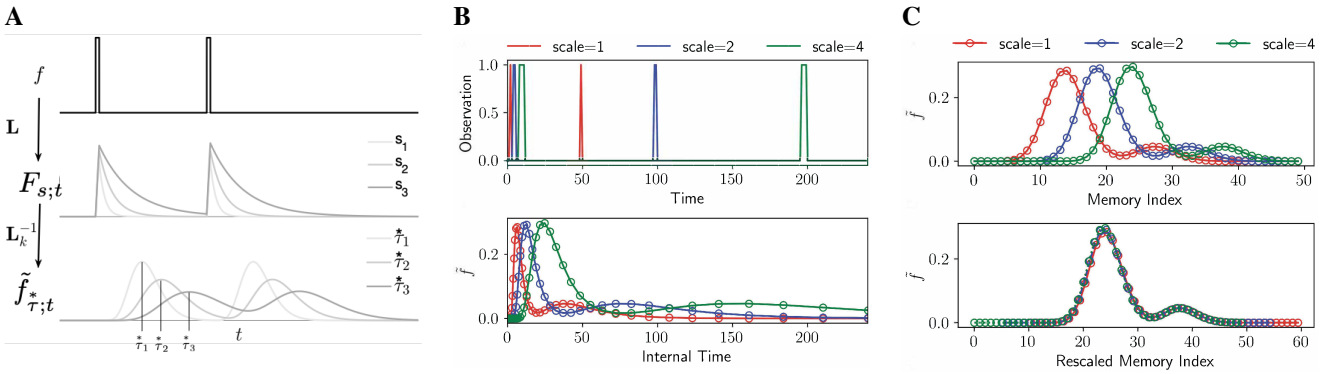

Figure 3: **A.** Response of the CogRNN to δ pulses. Neurons in $F_{s;t}$ decay exponentially at a spectrum of time constants $\mathbf{s}$ implementing a discrete approximation of a real-domain Laplace transform. Neurons in $\tilde{f}_{\overset{*}{\tau};s}$ activate sequentially, resembling time cells. **B.** Log-compressed memory (bottom) of three signals that are rescaled versions of each other (top) at time $t = 250$. Each circle represents the activity of individual neurons and their position along the x-axis corresponds to their peak time. **C.** Log-compressed memory turns rescaling into translation. The top plot is the same as the bottom plot in *B*, but with the x-axis corresponding to the neuron index instead of the peak time.

we designed a neural network architecture that maintains a scale invariant memory. Specifically, this network constructs an approximation of a real-domain Laplace transform of the temporal history of the input signal and then constructs an approximate inverse of the history – giving rise to an internal timeline of the past along a log-compressed axis characterized by sequentially activated neurons (Fig. 3A).

To construct a scale invariant memory we use a network composed of two layers. The input coming from the *encoder*, which we label as f, is fed into a recurrent layer (F) with the weights analytically computed to approximate the real-domain Laplace transform of the temporal history of the input. The output of the recurrent layer is mapped through a linear layer with analytically computed weights implementing the inverse Laplace transform $\tilde{f}$ (Fig. 3A). Below we describe this procedure step by step, first the continuous-time formulation and then discrete, neural network implementation.

Continuous-time formulation Given a one-dimensional input signal $f(t)$, we define a modified version of the Laplace transform $F(s;t)$:

$$F(s;t) = \int_0^t e^{-s(t-t')} f(t')dt'. \tag{1}$$

This modified version differs from the standard Laplace transform only in the variable s. Instead of s being a complex value composed of real and imaginary parts, we restrict s to a positive real value. This modification simplifies the neural network implementation while giving us the computational benefits of the standard Laplace transform, as illustrated below. Note that F is also a function of time t. This implies that at every moment, we construct the Laplace transform of the input function up to time t: $f(0 \le t' < t) \xrightarrow{L} F(s;t)$.

To construct the temporal history of the input, we need to invert the Laplace transform. The inverse, which we denote as $\tilde{f}(\overset{*}{\tau};t)$, can be computed using Post's inversion formula (Post 1930):

$$\tilde{f}(\overset{*}{\tau};t) = \mathbf{L}_k^{-1} F(s;t) = \frac{(-1)^k}{k!} s^{k+1} \frac{d^k}{ds^k} F(s;t), \tag{2}$$

where $\overset{*}{\tau} := k/s$ and $k \to \infty$ (see Tano, Dayan, and Pouget (2020); Horváth et al. (2020) for alternative approaches to computing the inverse transform).

Neural networks implementation To describe a neural network approximation of the Laplace transform, we first rewrite Eq. 1 in a differential form:

$$\frac{dF(s;t)}{dt} = -sF(s;t) + f(t). \tag{3}$$

The impulse response (response to input $f(t) = \delta(0)$) of $F(s;t)$ decays exponentially as a function of time t with decay rate s: e^{-st} (the second row in Fig. 3A). Note that this is a linear transform, so $F(s;t)$ will be a convolution between $f(t)$ and the impulse response.

We implement an approximation of the modified Laplace and inverse Laplace transform as a two-layer neural network with analytically computed weights. The first layer implements the modified Laplace transform through an RNN. The second layer implements the inverse Laplace transform as a dense layer with weights analytically calculated to compute a k-th order derivative with respect to s.

While in the Laplace domain s is a continuous variable, here we redefine s as a vector of N elements. We can now write a discrete-time approximation of Eq. 3 as an RNN with a diagonal connectivity matrix:

$$F_{s;t} = \mathbf{L}F_{s;t-1} + f_t, \tag{4}$$

where $\mathbf{L}$ is an $N \times N$ recurrent matrix $\mathbf{L} := e^{-\mathbf{S}\Delta t}$ implementing the discrete Laplace transform operator and $\mathbf{S}$ is a diagonal matrix composed of s values. At every time step t, $F_{s;t}$ is an N-element vector. For brevity of notation, we

assume that the duration of a discrete-time step $\Delta t = 1$. Following Eq. 2, a discrete approximation of the inverse Laplace transform, $\tilde{f}_{\overset{*}{\tau};t}$, can be implemented as a dense layer on top of $F_{s;t}$. The connectivity matrix of the dense layer is $\mathbf{L}_k^{-1}$(see Maini et al. (2023) for the derivation of the exact matrix form of $\mathbf{L}_k^{-1}$).

To interpret $\tilde{f}_{\overset{*}{\tau};t}$ and to select s values in an informed way, we compute the impulse response of $\tilde{f}_{\overset{*}{\tau};t}$. For input $f(t) = \delta(0)$, the activity of $\tilde{f}_{\overset{*}{\tau};t}$ is:

$$\tilde{f}_{\overset{*}{\tau};t} = \frac{1}{t}\frac{k^{k+1}}{k!}\left(\frac{t}{\overset{*}{\tau}}\right)^{k+1} e^{-k\frac{t}{\overset{*}{\tau}}}. \tag{5}$$

The impulse responses of units in $\tilde{f}_{\overset{*}{\tau};t}$ is a set of unimodal basis functions (Fig. 3A). To better characterize their properties, we first find the peak time by taking a partial derivative with respect to t, equate it with 0 and solve for t: $\partial \tilde{f}_{\overset{*}{\tau};t}/\partial t = 0 \rightarrow t = \overset{*}{\tau}$. Therefore, each unit in $\tilde{f}_{\overset{*}{\tau};t}$ peaks at $\overset{*}{\tau}$.

To further characterize our approximation, we express the width of the unimodal basis functions of the impulse response of $\tilde{f}_{\overset{*}{\tau};t}$ through the coefficient of variation c: $c = 1/\sqrt{k+1}$. Importantly, c does not depend on t and $\overset{*}{\tau}$, implying that the width of the unimodal basis functions increases linearly with their peak time. Therefore, when observed as a function of $\log(t)$, the width of the unimodal basis functions is constant.

We choose values of $\overset{*}{\tau}$ as log-spaced between some minimum $\overset{*}{\tau}_{min}$ and maximum $\overset{*}{\tau}_{max}$ as specified in the next section. Note that fixing the values of $\overset{*}{\tau}$ and choosing k also fixes values of s since $s = k/\overset{*}{\tau}$, so s is not a trainable parameter. Because of the log-spacing and because c does not depend on t and $\overset{*}{\tau}$, when analyzed as a function of $\log(t)$, the unimodal basis functions are equidistant and equally wide, providing uniform support over the $\log(t)$ axis (Fig 3C). Because the described system is linear, any input function is represented as a convolution with a set of these basis functions. As we demonstrate next, this produces a log-compressed memory that is scale invariant.

Invariance to temporal rescaling Following Eq. (2) and Eq. (5) we note that $\tilde{f}(\overset{*}{\tau};t)$ is scale invariant in the sense that rescaling $\tilde{f}(\overset{*}{\tau};t) \longrightarrow \tilde{f}(\overset{*}{\tau};at)$ can be undone by setting $\overset{*}{\tau}_i \longrightarrow \overset{*}{\tau}_i/a$. Choosing $\overset{*}{\tau}$ to be log-spaced ($\overset{*}{\tau}_i = (1+c)^{i-1}\overset{*}{\tau}_{min}$, with $c > 0$) makes the rescaling of $\overset{*}{\tau}$ equivalent to translation: $\overset{*}{\tau}_i = \overset{*}{\tau}_{i+\Delta}$ where $\Delta = log_{1+c}a$. This implies that temporal rescaling will cause a translation of the sequentially activated units (Fig. 3B,C).

This approach can be used to either build agents whose temporal memory representation covaries with the temporal scale (i.e., temporal rescaling is converted into translation as described above) or agents with temporal memory that is invariant of the temporal scale (i.e., if time in the environment rescales, the agent will have identical memory representation). When the objective is to build agents that are invariant to temporal rescaling, we apply convolution and pooling over $\tilde{f}$. The output of convolution and pooling is translation invariant, making the network invariant to temporal rescaling (top row in Fig. 5A). We note that since rescaling is converted into translation, the presence of edge effects impacts the output of convolution and pooling, thereby making the invariance imperfect.

Intuition To provide an intuitive understanding of the scale invariant memory, we refer to Fig. 3B. The top row shows three input signals each composed of two δ pulses. Each of the three signals is a rescaled version of another, i.e. $f_1(t) = f_2(a_1 t) = f_3(a_2 t)$ where a_i is the scaling factor ($a_1 = 2$ and $a_2 = 4$ in our example). The second row shows activity of neurons in the scale invariant memory of the above signals (this is the state of the memory at time $t = 250$). Neurons in a scale invariant memory are activated sequentially as a function of internal representation of time. Each circle in the plot represents activity of a single neuron and the x-axis corresponds to the log-spaced peak times (hence we refer to it as an internal representation time). Together, these neurons code a log-compressed memory of the input. This memory representation converts functions of time t into functions of $log(t)$. If time is rescaled by factor a, the resulting representation will be shifted (translated) by $\log(a)$: $f(at) \rightarrow f(\log(at)) = f(\log(a)+\log(t))$. Therefore, the memory representation is covariant with respect to rescaling. The strength of such a representation is that it makes computational problems that occur at different scales equally difficult. When plotted as a function of neuron index, rather than peak time (top row in Fig. 3C), the three signals corresponding to three temporal scales are now translated, rather than rescaled, versions of one another. If the x-axis is shifted by $\log(a)$, the three signals overlap (bottom row in Fig. 3C).

RL Agent

We used deep RL agents based on a synchronous version of A3C (Mnih et al. 2016) that uses advantage to calculate gradient-based policy updates. However, as the direct calculation of advantage may lead to high variance and longer training times, we have used an exponentially weighted estimator of advantage called generalized advantage estimator (GAE) (Schulman et al. 2016).

Hyperparameters and training

Our setup for the 3D interval timing environment used the three following convolutional layers: 32 kernels of size 8×8 (stride 2), 16 kernels of size 4×4 (stride 1), and 32 kernels of size 8×8 (stride 2). The fully connected layer after the convolution layer has 64 nodes. Outputs of all layers had ReLU activation. Following the encoder, the RL agent had either an LSTM network with 256 hidden units, or the CogRNN architecture with 8 log-spaced units having $\overset{*}{\tau}_{min} = 1$, $\overset{*}{\tau}_{max} = 1000$, and $k = 8$. We also introduced two layered multihead-attention (Vaswani et al. 2017) over

the $\overset{*}{\tau}$ with 8 heads and d_{model} 128. Parameters related to the RL algorithm were a discount factor (γ) of 0.98 and a decay (λ) factor of 0.95. We used the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1e^{-8}$. We also trained the RL agents with varying learning rates, including 0.001, 0.0001 and 0.00001, and selected the best-performing learning rates for each agent. We explored the impact of the hyperparameters (learning rate and entropy coefficient) and provided results as a part of the Supplemental Information.

For simple environments, we used LSTM networks with 128 hidden size and the same CogRNN network without the attention network. We updated the model parameters after each trial instead of backpropagating the gradients after horizon number of steps.[1]

Results

Agents with scale invariant memory learn equally well at different temporal scales

We first demonstrate that when temporal relationships between task-relevant variables rescale, CogRNN agents learn at about the same rate. This is because the log compression causes the memory representation to shift rather than rescale (Fig. 3B), making the learning problem equally difficult at every scale where the temporal relationships fall between the hyperparameters $\overset{*}{\tau}_{min}$ and $\overset{*}{\tau}_{max}$.

We trained agents based on CogRNN, LSTM and RNN in each of the five environments. To evaluate robustness to rescaling in four environments (interval timing 1D and 3D, interval discrimination and delayed-match-to-sample) we conducted the training with different durations of task-relevant temporal intervals. We controlled the duration using a step size parameter, which varied from 10 to 100. In interval timing for instance, with a step size of 10, the intervals ranged from 300 to 480 steps, while with a step size of 100, they ranged from 30 to 48 steps. All other parameters of the environment and of the agents remained unchanged across different scales.

The mean reward as a function of the number of trials in those four environments is shown in Fig. 4 for CogRNN and LSTM agents. Agents based on CogRNN reached high performance in all tasks. Critically, the speed of learning was similar at different temporal scales. While LSTM and RNN agents were able to learn in all environments except the 3D interval timing environment, they did so with different learning speeds for different temporal scales.

Psychometric curves shown in Fig. S2 provide another performance measure. The y-axis on the psychometric curves represents the probability of selecting the long interval. For an agent that performs perfectly, that probability would be zero for the three short intervals and one for the three long intervals. Consistent with the results in Fig. 4, CogRNN agents performed better than others. Similar to rat behavior, agents made the most mistakes for the most difficult time intervals (36 and 40 steps).

[1]Supplemental Information and code is available in https://github.com/cogneuroai/RL-with-scale-invariant-memory.

The results of training on the interval reproduction task are shown in Fig. S11, Fig. S12, Table S1, and Table S2. This task differs conceptually from the other four, since we did not train agents on rescaled versions of the environments. Instead, we trained on different interval durations, similar to Deverett et al. (2019). The results indicate that CogRNN agents were able to improve performance simultaneously at all training and validation intervals, demonstrating the capabilities of multi-scale learning. On the other hand, LSTM agents learned quickly only at short intervals. If LSTM agents were trained much longer, they could possibly learn the task at all scales, but critically, their learning is scale-dependent. We also highlight that training of CogRNN agents took roughly half the time as training of LSTM agents (since CogRNN has fewer trainable parameters).

Combining scale invariant memory with translation-invariance results in invariance to temporal rescaling

We now consider a case where the entire environment, including the duration of stimuli and the intervals between the stimuli, is rescaled in time (i.e., the elapse of time in the environment is sped up or slowed down, rather than just rescaling the time between task-relevant variables as in the previous case). Then combining scale invariant memory with convolution and maxpool results in agents which observation space is invariant (no longer covariant) of the temporal scale. If trained at one scale, such agents will follow the same policy when presented with the environment at a different scale and obtain the same performance as on the scale that was used for training (aside from edge effects mentioned in the Model section). The agents do not need to know the scale as long as the temporal relationships again fall between the hyperparameters $\overset{*}{\tau}_{min}$ and $\overset{*}{\tau}_{max}$.

We demonstrate this property by training agents in the 1D interval timing environment shown in Fig. 3B. For simplicity, instead of A2C, we used the REINFORCE algorithm (Williams 1992). Agents with scale invariant memory followed by convolution and pooling were able to learn the task after about 100k trials. We then rescaled the time in the observation space to 2x and 4x the initial scale. Without additional training, our agents reached perfect performance (Fig. 5B). This was not the case for agents trained with RNNs, as they failed to generalize across different temporal scales.

Neural activity is scale invariant and resembles time cells recorded from mammalian brains

Mammalian neural activity during tasks such as interval timing and delayed match to sample is characterized by neurons with monotonically growing/decaying firing rates and neurons that exhibit sequential activation (time cells) (Jin, Fujii, and Graybiel 2009; Tiganj et al. 2017, 2018). Consistent with the Weber-Fechner law, the width of the temporal windows in time cells increases with the peak time (Cao et al. 2022).

To test whether these properties from biological neurons are present in the artificial agents, we selected a representa-

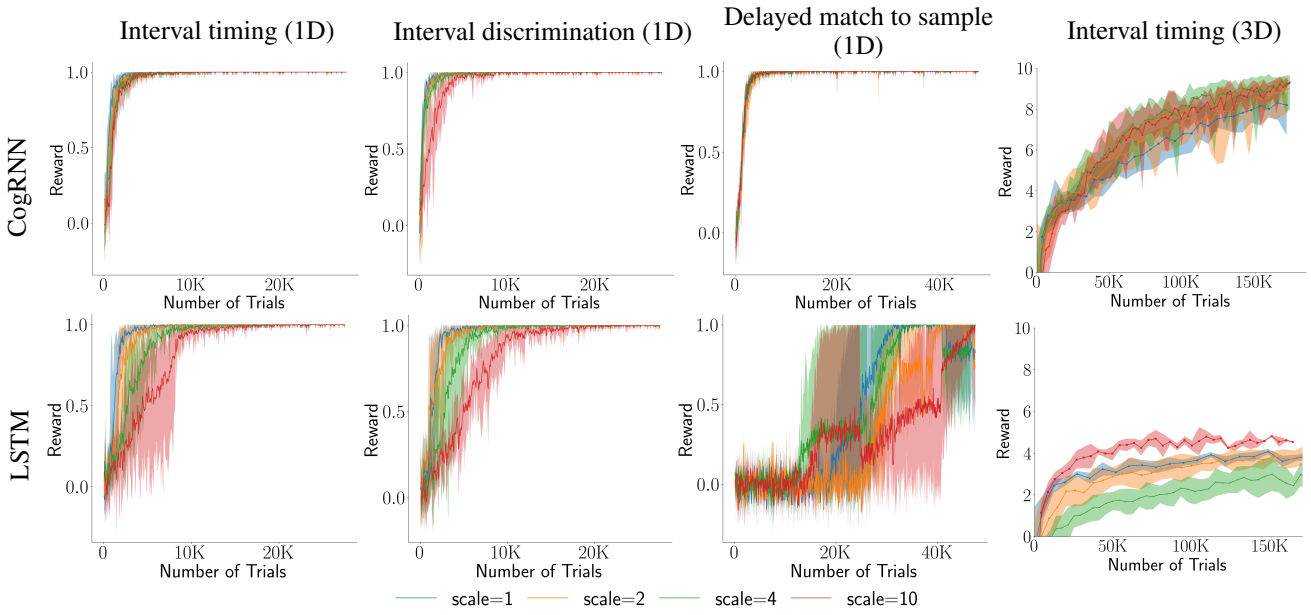

Figure 4: The performance (mean with standard error over five runs) across the four tasks for CogRNN and LSTM agents.

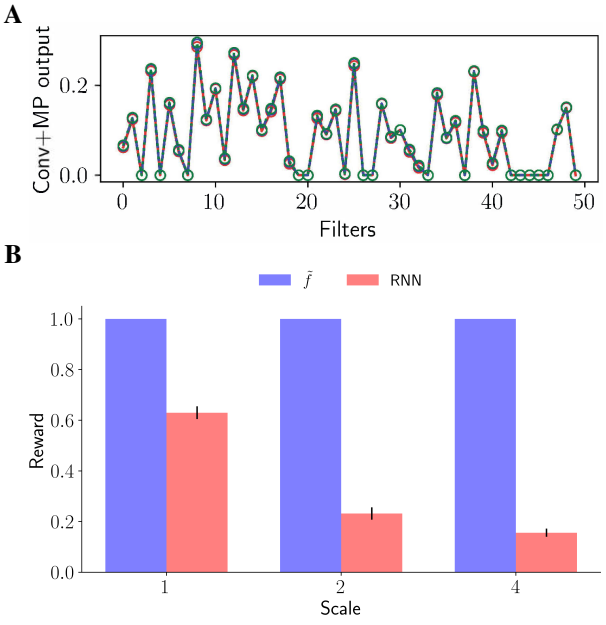

Figure 5: **A.** Output of convolution and pooling operations for three signals from Fig. 3B. **B.** Performance of CogRNN ($\tilde{f}$) and RNN agents trained on the 1D interval timing task. The agents were trained on scale 1 and evaluated on scales 1, 2 and 4.

tive agent from RNN, LSTM and CogRNN cores. To clean the data, for each of the three agents, we first removed neurons that were persistently active (i.e. that had constant activity, therefore not encoding any temporal information) and neurons that were completely silent. We then identified neurons that had monotonically growing/decaying activity, finding such neurons in all three agents (Fig. S3).

Next, we identified neurons with transient activation sometime during the timed interval (neurons that resemble time cells) and again found them in all three agents (Fig. 6). For CogRNN, the activation pattern resembles the scale invariant impulse responses shown in Fig. 3A.

To better understand the activation profiles, we fitted each time cell with a Gaussian distribution and estimated standard deviation. For a representation to be scale invariant, the relationship between the peak time and the standard deviation should be linear. This was the case only for CogRNN neurons (Fig. S4). Note that some of the CogRNN neurons with large peaks deviate from the linear relationship. This is because the numerical derivative d^k/ds^k suffers from edge effects. Increasing $\overset{*}{\tau}_{max}$ would extend the range of CogRNN representation while requiring a logarithmic increase in the number of neurons.

Ablation studies

To validate the robustness of our approach and understand better potential limitations, we ran several ablation studies that are added to the Supplemental Information. The ablations studies were conducted on the 3D interval timing environment. We trained the agents using LSTM and RNN with frozen weights, neither of which was able to learn the task (Fig. S1). Note that CogRNN can be considered a type of RNN with frozen (analytically calculated) weights.

We removed the inverse Laplace transform and instead of $\tilde{f}$ used CogRNN with F (the Laplace transform). CogRNN F agents were also not able to learn the task as well as $\tilde{f}$ (Fig. S1), indicating the importance of the inverse Laplace transform and the time cells (since those are generated through the inverse Laplace transform).

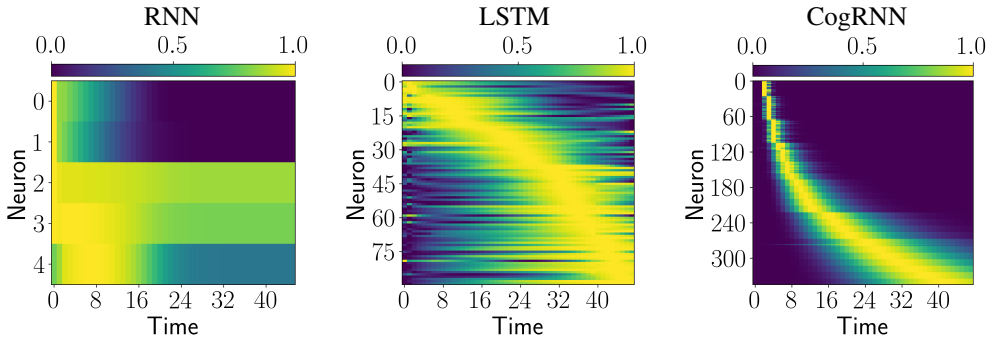

Figure 6: Normalized activity of neurons that resemble time cells from three representative agents. Neurons are sorted by peak time.

To examine the impact of CogRNN parameters we changed the values of $\overset{*}{\tau}_{min}$ (Fig. S5), $\overset{*}{\tau}_{max}$ (Fig. S6), and k (Fig. S7). In general, CogRNN $\tilde{f}$ agents reached high performance except in cases where hyperparameters were inappropriate for the properties of the environment.

We further evaluated the robustness of the results to the choice of the training parameters by varying learning rate (Fig. S8), entropy (Fig. S9) and horizon (Fig. S10). In general, for our main experiments, we selected the hyperparameters that led to the best results as specified in the Model section.

Discussion

Our results demonstrated that artificial agents can approach scale invariant learning when equipped with scale invariant memory. When temporal relationships in the environment rescale, the log compression in the scale invariant memory causes temporal memory to shift rather than rescale. This shift ensures the difficulty of learning the task remains consistent. We have demonstrated that when the environment is perfectly rescaled, the agents can generalize and use the knowledge acquired at one scale to solve the problem at different scales. These results were observed despite the fact that RL algorithms such as A2C contain components that break scale invariance, including exponential temporal discounting, horizon (n-step returns) and rollout length. Combining the proposed approach with scale invariant (power-law) temporal discounting (Tano, Dayan, and Pouget 2020; Tiganj et al. 2019; Redish and Kurth-Nelson 2010) could further improve the results.

We believe that the fundamental principles behind this work have the potential to impact research above and beyond interval timing, temporal discrimination and delayed match to sample. Humans can move objects from one place on the desk to another, furniture from one room to another, or drive a car from one location to another. We did not evolve to drive cars for hundreds of miles. But an ample amount of data from cognitive psychology suggests that we did evolve a scale invariant representation that enables us to efficiently and automatically represent arbitrary scales and operate on them. Building scale invariance into deep neural networks is critical for developing systems that can adjust to new environments rapidly and flexibly without the need for continual tweaking of hyperparameters.

Acknowledgments

This research was supported in part by Lilly Endowment, Inc., through its support for the Indiana University Pervasive Technology Institute.

References

Balci, F.; and Freestone, D. 2020. The peak interval procedure in rodents: a tool for studying the neurobiological basis of interval timing and its alterations in models of human disease. *Bio-protocol*, 10(17): e3735–e3735.

Balci, F.; and Simen, P. 2016. A decision model of timing. *Current opinion in behavioral sciences*, 8: 94–101.

Balsam, P. D.; and Gallistel, C. R. 2009. Temporal maps and informativeness in associative learning. *Trends in neurosciences*, 32(2): 73–78.

Banino, A.; Barry, C.; Uria, B.; Blundell, C.; Lillicrap, T.; Mirowski, P.; Pritzel, A.; Chadwick, M. J.; Degris, T.; Modayil, J.; et al. 2018. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705): 429–433.

Buhusi, C. V.; Aziz, D.; Winslow, D.; Carter, R. E.; Swearingen, J. E.; and Buhusi, M. C. 2009. Interval timing accuracy and scalar timing in C57BL/6 mice. *Behavioral neuroscience*, 123(5): 1102.

Buhusi, C. V.; and Meck, W. H. 2005. What makes us tick? Functional and neural mechanisms of interval timing. *Nature reviews neuroscience*, 6(10): 755–765.

Cao, R.; Bladon, J. H.; Charczynski, S. J.; Hasselmo, M. E.; and Howard, M. W. 2022. Internally generated time in the rodent hippocampus is logarithmically compressed. *Elife*, 11: e75353.

Coumans, E.; and Bai, Y. 2021. PyBullet, a Python module for physics simulation for games, robotics and machine learning. http://pybullet.org.

Cruzado, N. A.; Tiganj, Z.; Brincat, S. L.; Miller, E. K.; and Howard, M. W. 2020. Conjunctive representation of what and when in monkey hippocampus and lateral prefrontal

cortex during an associative memory task. *Hippocampus*, 30(12): 1332–1346.

Cueva, C. J.; Saez, A.; Marcos, E.; Genovesio, A.; Jazayeri, M.; Romo, R.; Salzman, C. D.; Shadlen, M. N.; and Fusi, S. 2020. Low-dimensional dynamics for working memory and time encoding. *Proceedings of the National Academy of Sciences*, 117(37): 23021–23032.

Deverett, B.; Faulkner, R.; Fortunato, M.; Wayne, G.; and Leibo, J. Z. 2019. Interval timing in deep reinforcement learning agents. *Advances in Neural Information Processing Systems*, 32.

Donnelly, N. A.; Paulsen, O.; Robbins, T. W.; and Dalley, J. W. 2015. Ramping single unit activity in the medial prefrontal cortex and ventral striatum reflects the onset of waiting but not imminent impulsive actions. *European Journal of Neuroscience*, 41(12): 1524–1537.

Eichenbaum, H. 2014. Time cells in the hippocampus: a new dimension for mapping memories. *Nature Reviews Neuroscience*, 15(11): 732–744.

Emmons, E. B.; De Corte, B. J.; Kim, Y.; Parker, K. L.; Matell, M. S.; and Narayanan, N. S. 2017. Rodent medial frontal control of temporal processing in the dorsomedial striatum. *Journal of Neuroscience*, 37(36): 8718–8733.

Fechner, G. 1860/1912. *Elements of Psychophysics. Vol. I.* Houghton Mifflin.

Gallistel, C. R.; and Gibbon, J. 2000. Time, rate, and conditioning. *Psychological Review*, 107(2): 289–344.

Gallistel, C. R.; and Shahan, T. A. 2024. Time-scale invariant contingency yields one-shot reinforcement learning despite extremely long delays to reinforcement. *Proceedings of the National Academy of Sciences*, 121(30): e2405451121.

Genovesio, A.; Tsujimoto, S.; and Wise, S. P. 2009. Feature- and Order-Based Timing Representations in the Frontal Cortex. *Neuron*, 63(2): 254–266.

Gibbon, J. 1977. Scalar expectancy theory and Weber's law in animal timing. *Psychological review*, 84(3): 279.

Gouvêa, T. S.; Monteiro, T.; Motiwala, A.; Soares, S.; Machens, C.; and Paton, J. J. 2015. Striatal dynamics explain duration judgments. *Elife*, 4: e11386.

Grossberg, S.; and Schmajuk, N. A. 1989. Neural dynamics of adaptive timing and temporal discrimination during associative learning. *Neural networks*, 2(2): 79–102.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.

Horváth, G.; Horváth, I.; Almousa, S. A.-D.; and Telek, M. 2020. Numerical inverse Laplace transformation using concentrated matrix exponential distributions. *Performance Evaluation*, 137: 102067.

Howard, M. W.; MacDonald, C. J.; Tiganj, Z.; Shankar, K. H.; Du, Q.; Hasselmo, M. E.; and Eichenbaum, H. 2014. A unified mathematical framework for coding time, space, and sequences in the hippocampal region. *Journal of Neuroscience*, 34(13): 4692–4707.

Howard, M. W.; Shankar, K. H.; Aue, W. R.; and Criss, A. H. 2015. A distributed representation of internal time. *Psychological review*, 122(1): 24.

Jacques, B.; Tiganj, Z.; Howard, M. W.; and Sederberg, P. B. 2021. DeepSITH: Efficient Learning via Decomposition of What and When Across Time Scales. *Advances in neural information processing system*.

Jacques, B. G.; Tiganj, Z.; Sarkar, A.; Howard, M.; and Sederberg, P. 2022. A deep convolutional neural network that is invariant to time rescaling. In *International Conference on Machine Learning*, 9729–9738. PMLR.

Jazayeri, M.; and Shadlen, M. N. 2015. A neural mechanism for sensing and reproducing a time interval. *Current Biology*, 25(20): 2599–2609.

Jin, D. Z.; Fujii, N.; and Graybiel, A. M. 2009. Neural representation of time in cortico-basal ganglia circuits. *Proceedings of the National Academy of Sciences*, 106(45): 19156–19161.

Kim, J.; Ghim, J.-W.; Lee, J. H.; and Jung, M. W. 2013. Neural correlates of interval timing in rodent prefrontal cortex. *Journal of Neuroscience*, 33(34): 13834–13847.

Kim, Y.-C.; Han, S.-W.; Alberico, S. L.; Ruggiero, R. N.; De Corte, B.; Chen, K.-H.; and Narayanan, N. S. 2017. Optogenetic stimulation of frontal D1 neurons compensates for impaired temporal control of action in dopamine-depleted mice. *Current biology*, 27(1): 39–47.

Lin, D.; and Richards, B. A. 2021. Time cell encoding in deep reinforcement learning agents depends on mnemonic demands. *bioRxiv*, 2021–07.

Ludvig, E. A.; Sutton, R. S.; and Kehoe, E. J. 2008. Stimulus representation and the timing of reward-prediction errors in models of the dopamine system. *Neural computation*, 20(12): 3034–3054.

MacDonald, C. J.; Lepage, K. Q.; Eden, U. T.; and Eichenbaum, H. 2011. Hippocampal "time cells" bridge the gap in memory for discontiguous events. *Neuron*, 71(4): 737–749.

MacDonald, C. J.; and Tonegawa, S. 2021. Crucial role for CA2 inputs in the sequential organization of CA1 time cells supporting memory. *Proceedings of the National Academy of Sciences*, 118(3): e2020698118.

Maini, S. S.; Mochizuki-Freeman, J.; Indi, C. S.; Jacques, B. G.; Sederberg, P. B.; Howard, M. W.; and Tiganj, Z. 2023. Representing latent dimensions using compressed number lines. In *2023 international joint conference on neural networks (ijcnn)*, 1–10. IEEE.

Masset, P.; Tano, P.; Kim, H. R.; Malik, A. N.; Pouget, A.; and Uchida, N. 2023. Multi-timescale reinforcement learning in the brain. *bioRxiv*.

Matell, M. S.; and Meck, W. H. 2004. Cortico-striatal circuits and interval timing: coincidence detection of oscillatory processes. *Cognitive brain research*, 21(2): 139–170.

Mello, G. B.; Soares, S.; and Paton, J. J. 2015. A scalable population code for time in the striatum. *Current Biology*, 25(9): 1113–1122.

Miller, E. K.; Erickson, C. A.; and Desimone, R. 1996. Neural Mechanisms of Visual Working Memory in Prefrontal

Cortex of the Macaque. *Journal of Neuroscience*, 16(16): 5154–5167.

Mnih, V.; Badia, A. P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; and Kavukcuoglu, K. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, 1928–1937. PMLR.

Mochizuki-Freeman, J.; Kabir, M. R.; and Tiganj, Z. 2024. Incorporating a cognitive model for evidence accumulation into deep reinforcement learning agents. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 46.

Molano-Mazon, M.; Barbosa, J.; Pastor-Ciurana, J.; Fradera, M.; Zhang, R.-Y.; Forest, J.; del Pozo Lerida, J.; Ji-An, L.; Cueva, C. J.; de la Rocha, J.; et al. 2022. NeuroGym: An open resource for developing and sharing neuroscience tasks.

Momennejad, I.; and Howard, M. W. 2018. Predicting the future with multi-scale successor representations. *bioRxiv*, 449470.

Namboodiri, V. M. K. 2022. How do real animals account for the passage of time during associative learning? *Behavioral Neuroscience*.

Narayanan, N. S. 2016. Ramping activity is a cortical mechanism of temporal control of action. *Current opinion in behavioral sciences*, 8: 226–230.

O'Keefe, J. 1976. Place units in the hippocampus of the freely moving rat. *Experimental neurology*, 51(1): 78–109.

Parker, K. L.; Chen, K.-H.; Kingyon, J. R.; Cavanagh, J. F.; and Narayanan, N. S. 2014. D1-dependent 4 Hz oscillations and ramping activity in rodent medial frontal cortex during interval timing. *Journal of Neuroscience*, 34(50): 16774–16783.

Pastalkova, E.; Itskov, V.; Amarasingham, A.; and Buzsaki, G. 2008. Internally generated cell assembly sequences in the rat hippocampus. *Science*, 321(5894): 1322–1327.

Pérez, O.; and Merchant, H. 2018. The synaptic properties of cells define the hallmarks of interval timing in a recurrent neural network. *Journal of Neuroscience*, 38(17): 4186–4199.

Petter, E. A.; Gershman, S. J.; and Meck, W. H. 2018. Integrating models of interval timing and reinforcement learning. *Trends in cognitive sciences*, 22(10): 911–922.

Post, E. 1930. Generalized Differentiation. *Transactions of the American Mathematical Society*, 32: 723–781.

Raphan, T.; Dorokhin, E.; and Delamater, A. R. 2019. Modeling interval timing by recurrent neural nets. *Frontiers in integrative neuroscience*, 13: 46.

Redish, A. D.; and Kurth-Nelson, Z. 2010. Neural models of delay discounting.

Salz, D. M.; Tiganj, Z.; Khasnabish, S.; Kohley, A.; Sheehan, D.; Howard, M. W.; and Eichenbaum, H. 2016. Time cells in hippocampal area CA3. *Journal of Neuroscience*, 36(28): 7476–7484.

Schaeffer, R.; Khona, M.; and Fiete, I. 2022. No free lunch from deep learning in neuroscience: A case study through models of the entorhinal-hippocampal circuit. *bioRxiv*, 2022–08.

Schulman, J.; Moritz, P.; Levine, S.; Jordan, M. I.; and Abbeel, P. 2016. High-Dimensional Continuous Control Using Generalized Advantage Estimation. In Bengio, Y.; and LeCun, Y., eds., *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Shankar, K. H.; and Howard, M. W. 2012. A scale-invariant internal representation of time. *Neural Computation*, 24(1): 134–193.

Shankar, K. H.; and Howard, M. W. 2013. Optimally fuzzy temporal memory. *Journal of Machine Learning Research*, 14: 3753–3780.

Sorscher, B.; Mel, G.; Ganguli, S.; and Ocko, S. 2019. A unified theory for the origin of grid cells through the lens of pattern formation. *Advances in neural information processing systems*, 32.

Tano, P.; Dayan, P.; and Pouget, A. 2020. A Local Temporal Difference Code for Distributional Reinforcement Learning. *Advances in Neural Information Processing Systems*, 33.

Tiganj, Z.; Cromer, J. A.; Roy, J. E.; Miller, E. K.; and Howard, M. W. 2018. Compressed timeline of recent experience in monkey lateral prefrontal cortex. *Journal of cognitive neuroscience*, 30(7): 935–950.

Tiganj, Z.; Gershman, S. J.; Sederberg, P. B.; and Howard, M. W. 2019. Estimating scale-invariant future in continuous time. *Neural computation*, 31(4): 681–709.

Tiganj, Z.; Jung, M. W.; Kim, J.; and Howard, M. W. 2017. Sequential firing codes for time in rodent medial prefrontal cortex. *Cerebral Cortex*, 27(12): 5663–5671.

Tiganj, Z.; Tang, W.; and Howard, M. 2021. A computational model for simulating the future using a memory timeline. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 43.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017. Attention is All you Need. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Wang, J.; Narain, D.; Hosseini, E. A.; and Jazayeri, M. 2018. Flexible timing by temporal scaling of cortical responses. *Nature neuroscience*, 21: 102–110.

Whittington, J. C.; Warren, J.; and Behrens, T. E. 2021. Relating transformers to models and neural representations of the hippocampal formation. *arXiv preprint arXiv:2112.04035*.

Wilkes, J. T. 2015. *Reverse first principles: Weber's law and optimality in different senses*. Ph.D. thesis, UNIVERSITY OF CALIFORNIA, SANTA BARBARA.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8: 229–256.

Supplemental Information

As Supplemental Information, we provide additional results on the 3D interval timing environment:

- Results of the ablation analysis where agents were trained with frozen RNN, frozen LSTM, and CogRNN F (Fig. S1). None of the agents reached good performance.
- Psychometric curves (Fig. S2). Similar to rat behavior, agents made the most mistakes for the most difficult time intervals (36 and 40 steps).
- Activity heatmaps for neurons with monotonically decreasing/increasing activity (Fig. S3). CogRNN $\tilde{f}$ neurons resemble sequentially activated time cells recorded in mammalian brains where width increases with the peak time (Fig. S4).
- Results of hyperparameter exploration with different values of:
  - $\overset{*}{\tau}_{min}$ (Fig. S5),
  - $\overset{*}{\tau}_{max}$ (Fig. S6),
  - k (Fig. S7),
  - learning rate (Fig. S8),
  - entropy coefficient (Fig. S9),
  - horizon (Fig. S10).
- Results of training on interval reproduction task (Table S1, Table S2, Fig. S11, Fig. S12).

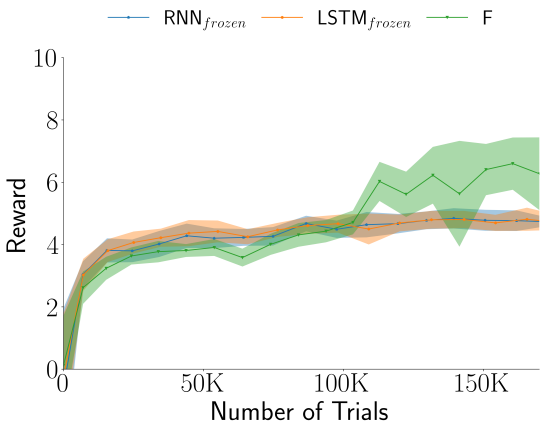

Figure S1: Performance of the frozen RNN, frozen LSTM, and CogRNN F agents for the same time scale during the 3D interval timing task.

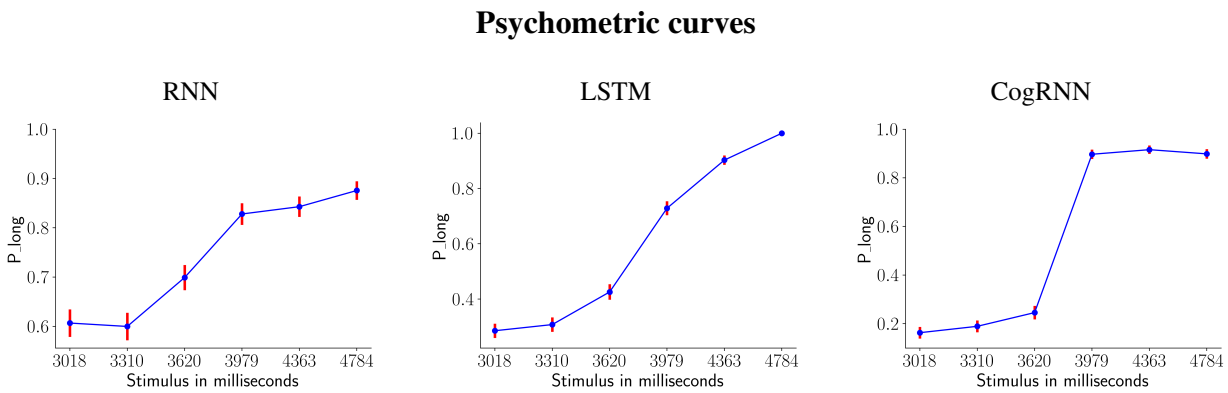

Figure S2: Psychometric curves for (a) RNN, (b) LSTM, and (c) CogRNN $\tilde{f}$ agents during 3D interval timing task. The average performance across agents is shown for each group, with error bars indicating the confidence intervals.

Neural activity in RNN, LSTM, and CogRNN agents in 3D interval timing task

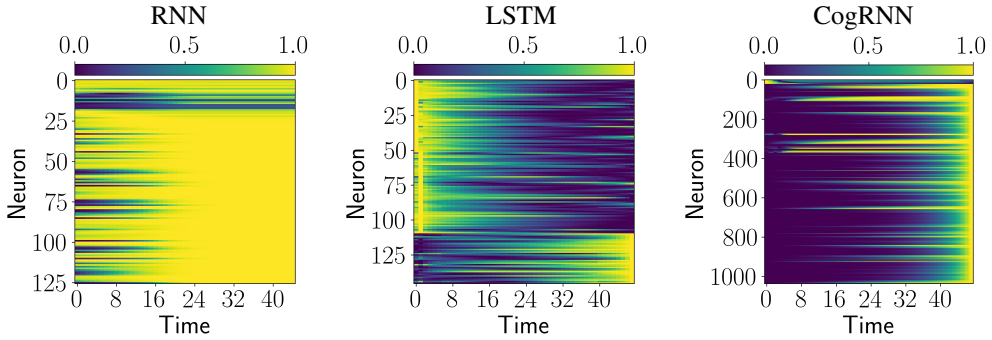

Figure S3: Normalized activity of neurons that have monotonically decreasing/increasing activity from three representative agents. Neurons are sorted by peak time.

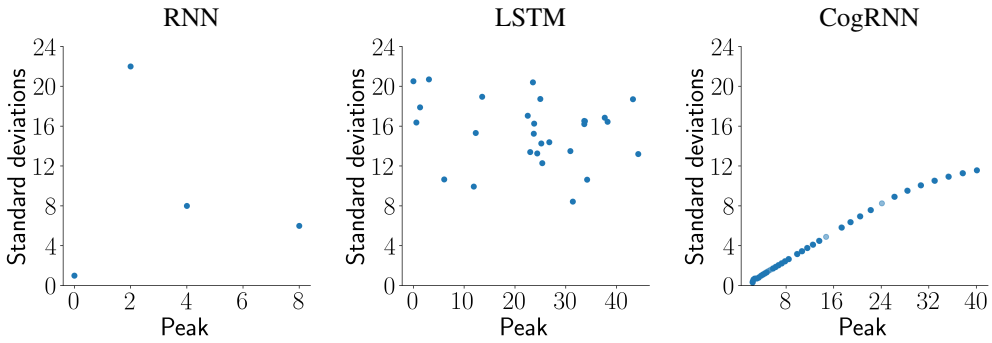

Figure S4: Relationship between peak time and standard deviation for neurons whose activity resembled time cells.

Hyperparameter exploration for CogRNN in the 3D interval timing task

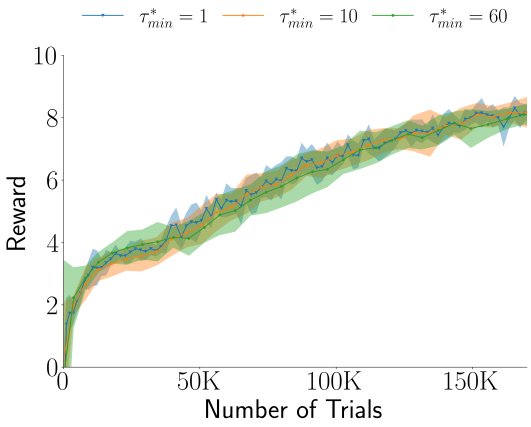

Figure S5: Performances of the CogRNN $\tilde{f}$ agents for different values of $\overset{*}{\tau}_{min}$.

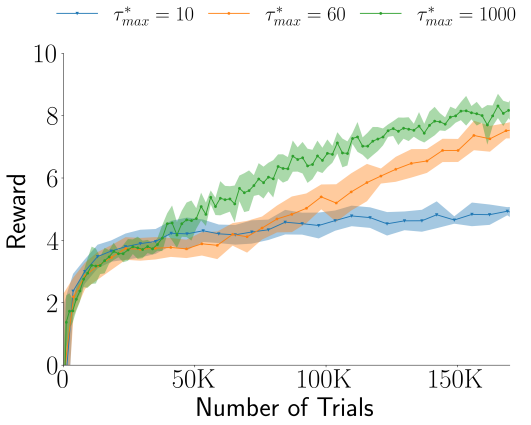

Figure S6: Performances of the CogRNN $\tilde{f}$ agents for different values of $\overset{*}{\tau}_{max}$.

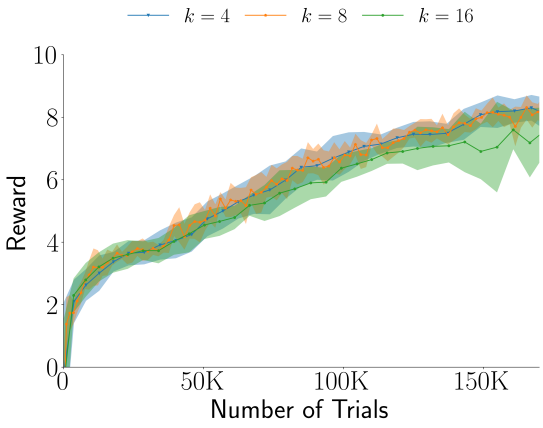

Figure S7: Performances of the CogRNN $\tilde{f}$ agents for different values of k.

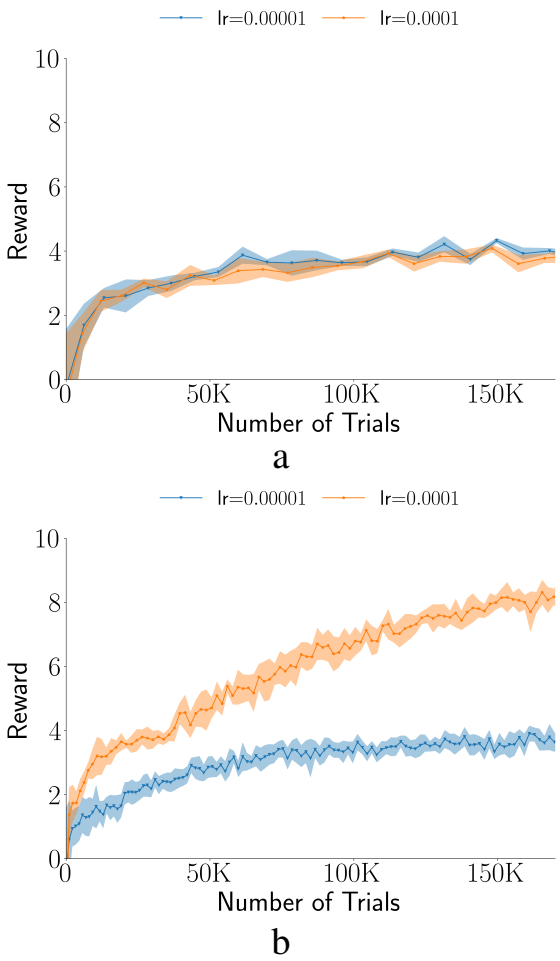

Figure S8: Performances of (a) LSTM and (b) CogRNN $\tilde{f}$ agents for different learning rates.

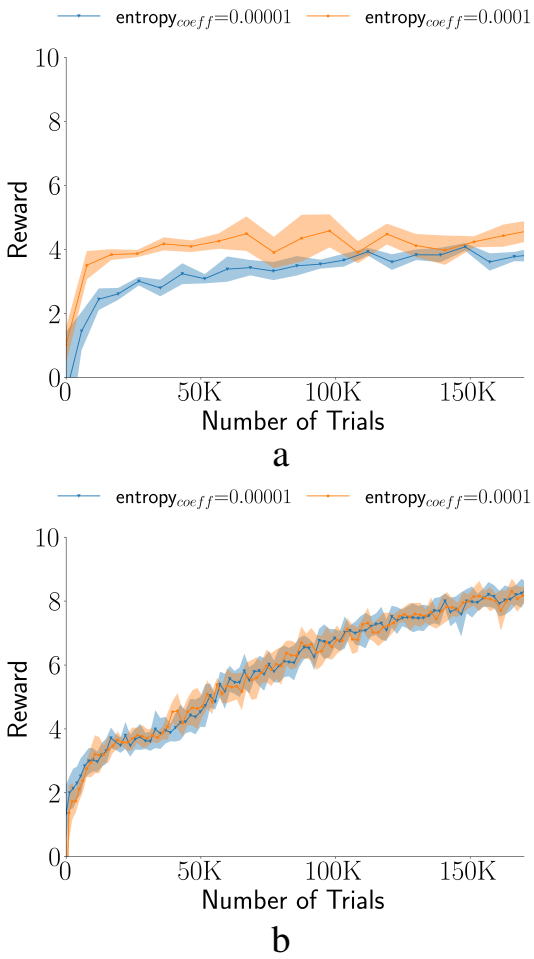

Figure S9: Performances of (a) LSTM and (b) CogRNN $\tilde{f}$ agents for different entropy coefficients.

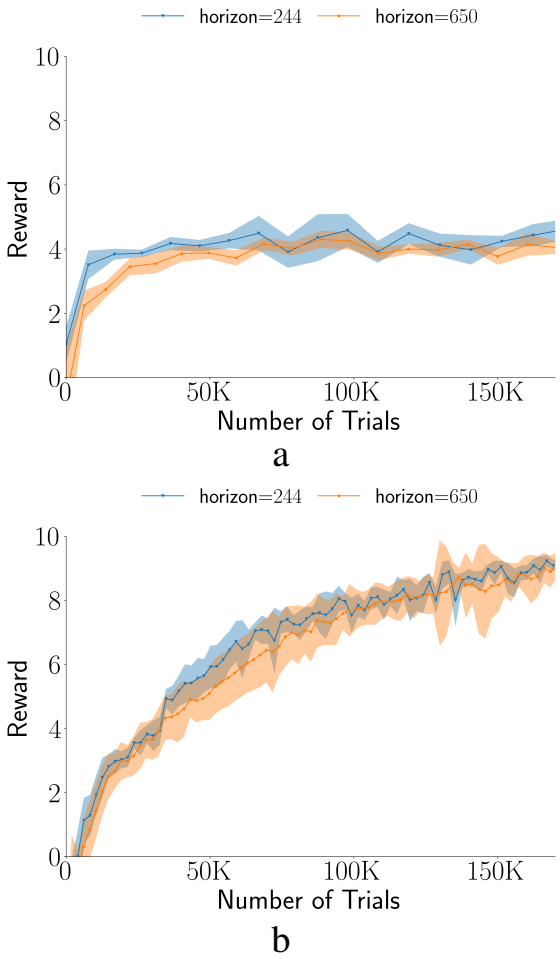

Figure S10: Performances of (a) LSTM and (b) CogRNN $\tilde{f}$ agents for different horizon lengths.

Results of training on Interval Reproduction task

Trial	Training intervals									
	10	20	30	40	50	60	70	80	90	100
25k	.8±.4	.5±.4	.5±.4	.6±.4	.5±.3	.8±.2	.9±.1	.7±.3	.6±.2	.2±.1
50k	.8±.4	.6±.5	.8±.4	.8±.4	.7±.4	1.0±.0	1.0±.0	1.0±.0	.9±.1	.9±.1
100k	1.0±.0	.6±.5	.8±.4	.8±.4	1.0±.0	1.0±.0	1.0±.0	.9±.1	.8±.4	.6±.4

Trial	Validation intervals								
	15	25	35	45	55	65	75	85	95
25k	.2±.2	.5±.4	.6±.4	.5±.3	.6±.3	.7±.2	.9±.2	.6±.2	.4±.2
50k	.2±.3	.5±.5	.8±.4	.7±.4	.9±.2	1.0±.1	.9±.1	1.0±.1	.9±.1
100k	.2±.3	.6±.5	.7±.4	.8±.3	1.0±.0	1.0±.1	1.0±.0	.7±.4	.8±.4

Table S1: Mean accuracy and standard-deviation in learning the interval reproduction task for the CogRNN agent.

Trial	Training intervals									
	10	20	30	40	50	60	70	80	90	100
25k	1.0±.0	.1±.1	.1±.1	.2±.1	.2±.1	.1±.0	.1±.0	.0±.1	.1±.1	.1±.1
50k	.4±.4	.2±.2	.2±.2	.1±.1	.1±.1	.3±.1	.3±.1	.1±.1	.1±.1	.1±.1
100k	1.0±.0	.4±.4	.4±.3	.2±.1	.2±.1	.2±.1	.2±.1	.1±.1	.1±.1	.1±.1

Trial	Validation intervals								
	15	25	35	45	55	65	75	85	95
25k	.2±.2	.1±.1	.1±.0	.2±.1	.2±.1	.1±.1	.2±.2	.2±.2	.3±.1
50k	.1±.1	.2±.3	.2±.1	.1±.1	.1±.1	.2±.1	.2±.1	.1±.0	.1±.1
100k	.6±.4	.3±.4	.3±.2	.2±.1	.1±.1	.1±.1	.1±.0	.1±.1	.1±.1

Table S2: Mean accuracy and standard-deviation in learning the interval reproduction task for the LSTM agent.

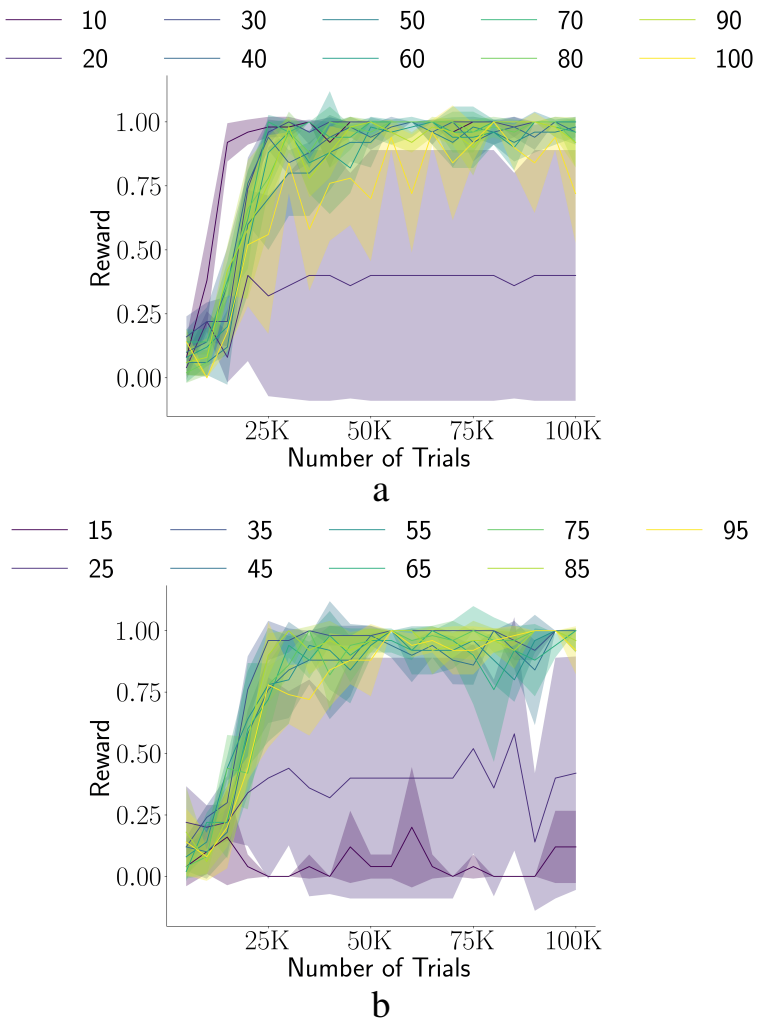

Figure S11: Performances of CogRNN $\tilde{f}$ agents in learning (a) train and (b) validation intervals for the interval reproduction task.

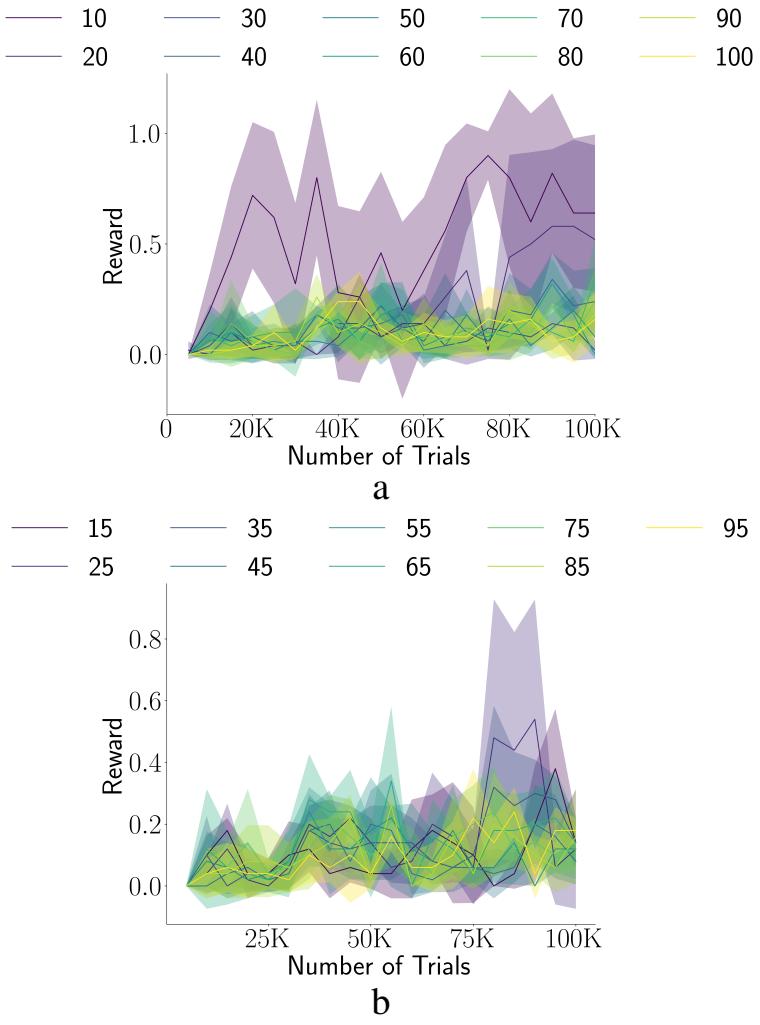

Figure S12: Performances of LSTM agents in learning (a) train and (b) validation intervals for the interval reproduction task.