PREDICTIVE LEARNING ENABLES NEURAL NETWORKS TO LEARN COMPLEX WORKING MEMORY TASKS

Thijs L. van der Plas
Department of Physiology, Anatomy and Genetics
University of Oxford
United Kingdom
thijs.vanderplas@dpag.ox.ac.uk

Tim P. Vogels*
Institute of Science and Technology Austria
Austria
tim.vogels@ist.ac.at

Sanjay G. Manohar*
Nuffield Department of Clinical Neurosciences
University of Oxford
United Kingdom
sanjay.manohar@psy.ox.ac.uk

ABSTRACT

Brains are thought to engage in predictive learning - learning to predict upcoming stimuli - to construct an internal model of their environment. This is especially notable for spatial navigation, as first described by Tolman's latent learning tasks. However, predictive learning has also been observed in sensory cortex, in settings unrelated to spatial navigation. Apart from normative frameworks such as active inference or efficient coding, what could be the utility of learning to predict the patterns of occurrence of correlated stimuli? Here we show that prediction, and thereby the construction of an internal model of sequential stimuli, can bootstrap the learning process of a working memory task in a recurrent neural network. We implemented predictive learning alongside working memory match-tasks, and networks emerged to solve the prediction task first by encoding information across time to predict upcoming stimuli, and then eavesdropped on this solution to solve the matching task. Eavesdropping was most beneficial when neural resources were limited. Hence, predictive learning acts as a general neural mechanism to learn to store sensory information that can later be essential for working memory tasks.

1 INTRODUCTION

Animals learn to create internal representations of their environment, even if these are not directly associated with attaining rewards (Tolman, 1948). How might an organism develop an internal model of their environment in the absence of reward? A candidate mechanism is predictive learning (Bar, 2009; Friston et al., 2016; Alexander & Brown, 2018; Nagai, 2019; Whittington et al., 2020; Recanatesi et al., 2021) which may arise even through passive exposure (Nagai, 2019). In predictive learning, animals continuously attempt to predict their future state, given their current state and actions. The utility of such models that predict future states has been extensively studied in reinforcement learning. In contrast to 'model-free' agents (that only reinforce actions that have previously led to rewards), 'model-based' agents can rapidly adapt to changes in the environment or reward structure, instead of starting from scratch (Daw et al., 2011; Russek et al., 2017; Wayne et al., 2018). Further, predictive learning in recurrent neural networks (RNNs) can extract the latent structure of the environment during a period of unrewarded exploration, actualizing the proposal that predictive learning acts as a helper function in the brain by encoding the environment (Whittington et al., 2020; Recanatesi et al., 2021).

Neuroscience experiments have revealed that passive learning of the environment may indeed encode spatial structure in the brain, as observed in the hippocampus (O'Keefe & Dostrovsky, 1971), but also nonspatial sequences of perceptual states, which can be detected even in primary sensory cortex (Libby & Buschman, 2021). However, in a purely passive learning paradigm, it remains unclear what benefit an organism would gain from actively maintaining predictions of upcoming sensory inputs. To address this question, we focus on a working memory task, which specifically requires maintaining information across time, and ask whether actively maintaining sensory predictions across a time delay can improve learning.

* Co-senior author

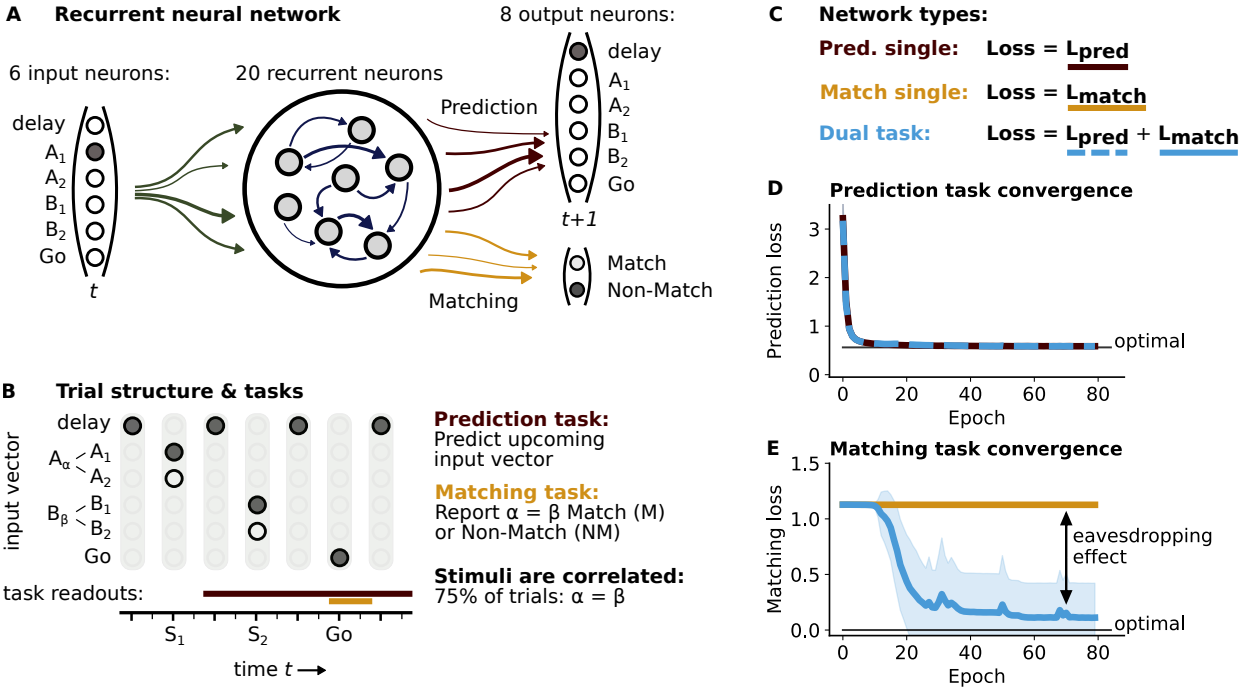

Figure 1: **RNNs use eavesdropping to bootstrap learning.** **A)** RNNs projected to two sets of output neurons, corresponding to the prediction task and matching task. **B)** Each trial consisted of two stimuli S_1 and S_2 followed by a GO response window, interleaved by delay periods. The stimuli take the possible values $S_1 = A_\alpha \in \{A_1, A_2\}$ and $S_2 = B_\beta \in \{B_1, B_2\}$, where $P(\alpha = 1) = P(\alpha = 2) = 0.5$ and $P(\alpha = \beta) = 0.75$ (hence S_1 is predictive of S_2). **C)** Networks were trained with one of three possible loss functions: prediction single task learning (red), matching single task learning (orange) or dual task learning (blue). The dual task loss function was defined as the sum of the prediction task and matching task loss functions. **D)** Both prediction single task networks and dual task networks learned the prediction task equivalently well, converging to the optimal solution equally quickly. **E)** Only dual task networks could learn the matching task, after they solved the prediction task (see panel D). This sequence of task learning was not specified, as the loss function was set to be the sum of the loss functions of both tasks (panel C). Panels D and E show the mean $\pm$ std of 20 simulated networks per network type.

We use artificial neural networks to show that predictive learning can give rise to an internal stimulus representation that enables networks to learn a more difficult working memory task. Specifically, in our experiments, small, sparse RNNs could not learn a delayed-match-to-category (DMC) task without simultaneously engaging in predictive learning. When RNNs were faced with these two tasks, they would first solve the prediction task, which created an internal stimulus representation that kick-started learning of the DMC task. This phenomenon - where two tasks depend on a latent feature that can only be learned from one of the tasks, thereby initiating learning of the other - has been termed 'eavesdropping' (Caruana, 1997), and constitutes a form of multitask learning that is known to boost learning performance in artificial networks (Bell & Renals, 2015; Yu & Jiang, 2016; Liu et al., 2020). Our results suggest that neural systems could engage in eavesdropping by using predictive learning as helper task.

We also show that predictability does not need to persist constantly, in order to produce long-term benefits. It was sufficient to introduce stimulus predictability only transiently, making simulated annealing an effective mechanism to kick-start the learning of complex concepts. This finding aligns with the dynamic nature of natural environments, where stimulus correlations come and go. Finally, we leverage our results to formulate testable predictions for experimental neuroscience.

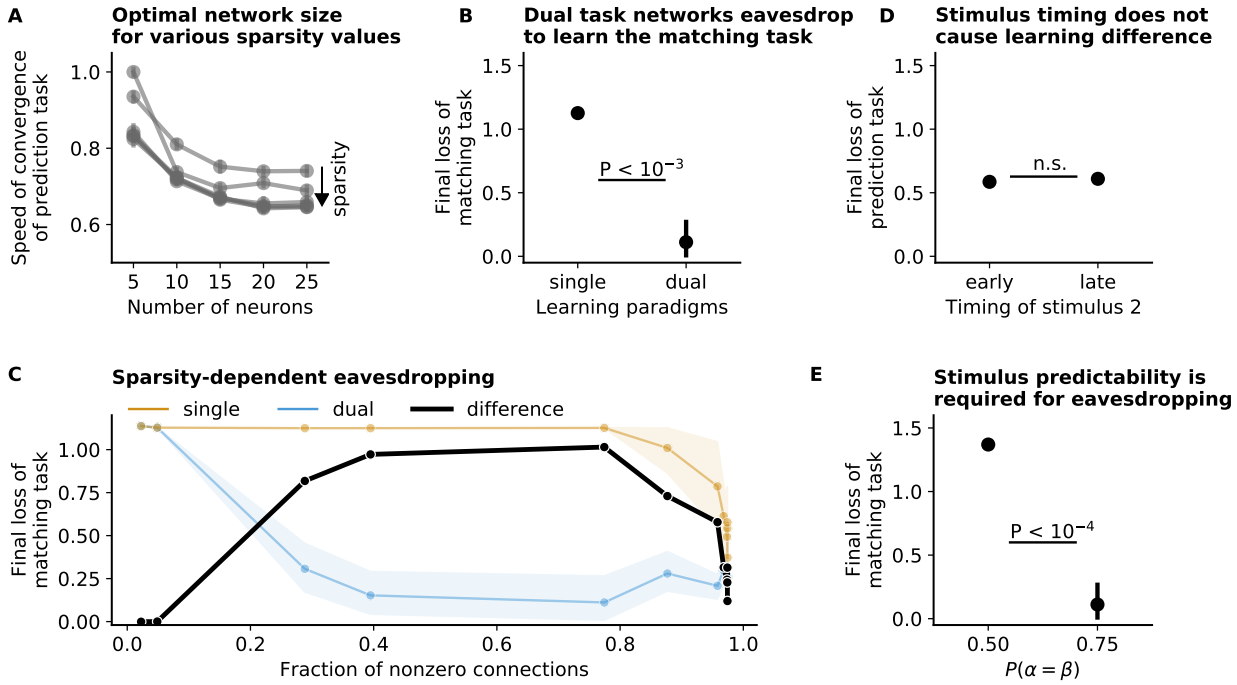

Figure 2: **Quantification of the eavesdropping effect A)** Speed of convergence of the prediction task as a function of network size and sparsity. **B)** The eavesdropping effect shown in Fig. 1E was quantified by measuring the difference in final loss (averaged over 5 final epochs), confirming that only dual task networks could learn the matching task. **C)** The magnitude of the eavesdropping effect depended on the number of nonzero connections (defined by $|W| > 0.01$, result is robust to different thresholds). The number of nonzero connections was varied by parametrically changing the sparsity regularization parameter λ of Eq. 3 from 0 to 0.1 and measuring the fraction of nonzero connections. **D)** The prediction task was solved equally well when S_2 was delivered later, during the GO window. **E)** Eavesdropping only occurred when the stimuli were predictable ($P(\alpha = \beta) = 0.75$), as in prior panels). Panels B, D and E used a two-sided Wilcoxon signed-rank test to assess significance (P values shown). All panels show the mean $\pm$ 95% confidence interval of 10 simulated networks per data point for panel A and 20 simulated networks per data point of all other panels (error bars or shaded area).

2 Results

2.1 Recurrent neural networks use eavesdropping to bootstrap learning

We trained standard recurrent neural networks (RNNs) to solve two working-memory tasks, either individually or simultaneously (Fig. 1A). Trials consisted of a sequence of two stimuli and a GO cue, interleaved by delay periods. The prediction task asked networks to continuously predict the next upcoming input stimuli. The matching task prompted during the GO period whether the two previously presented stimuli matched (Fig. 1B). Importantly, the stimuli were predictable, or biased, because a match occurred on 75% of trials. This means that the first stimulus S_1 was predictive of the second stimulus S_2 on most trials. Notably, networks could only learn the matching task when it was presented jointly with the prediction task (Fig. 1C, D, E). Here, networks first solved the prediction task, after which learning of the matching task ensued. As we will detail in the next sections, this happened because the prediction task unveiled a latent feature - the memory of the first stimulus S_1 - which was required to solve the matching task. This phenomenon, in which two tasks share a feature that can (easily) be learned from one, thus catalyzing learning in the second, is a multitask learning mechanism called 'eavesdropping' (Caruana, 1997).

2.2 Quantification of the eavesdropping effect

To quantify the eavesdropping effect (Fig. 1E) and explore what conditions facilitated this learning strategy, we first optimized the number of RNN hidden units based on their performance on the prediction task alone (Fig. 2A). We

found that networks with 20 neurons converged on the prediction task with near-maximal speed of convergence. We then assessed the eavesdropping effect by calculating the difference in performance (i.e., their final loss value after training) between matching single task networks (Fig. 1C, E, in orange) and dual task networks (Fig. 1C, E, in blue), averaged over 20 simulations of each (Fig. 2B).

Next, we quantified how the eavesdropping effect depended on the difficulty of the tasks, by varying the strength of sparsity regularization (Fig. 2C). We found that eavesdropping was most effective at a range of medium sparsity values from 30% to 80% non-zero connections. When sparsity regularization was too strong (i.e., less than 25% of all possible connections were allowed), both learning paradigms failed to learn. When sparsity regularization was very weak (i.e., more than 80% of all connections were allowed) both paradigms partially converged. Hence, the effectiveness of eavesdropping depended on the difficulty of the tasks, and eavesdropping allowed sparse networks to learn the matching task which is typically beneficial to prevent overfitting. Correspondingly, the best absolute performance on the matching task was only achieved with sparse networks that used eavesdropping (Fig. 2C). We performed this analysis across a range of network sizes N, which resulted in an equivalent range of sparsity values where networks could only learn the matching task with eavesdropping (Fig. S1). For the remainder of this study, we used the optimal fraction of nonzero connections (77%) from Fig. 2C.

Networks had to create an internal representation, i.e., a memory, of S_1 to predict the upcoming stimulus S_2. For the matching task, networks also had to memorize S_1 (to compare the two stimuli), but for a longer period of time (to compare with S_2). Therefore, the difference in performance (Fig. 2B) could in theory be explained by vanishing gradients. However, we found that this is not the case, as the prediction task was equally well solved when the delay period between S_1 and S_2 was increased to mirror the matching task (Fig. 2D).

Finally, as expected, we found that stimulus predictability $P(\alpha = \beta) > 0.5$ was crucial for eavesdropping (on the prediction task) to occur: dual task networks failed to learn the matching task when they were trained with uncorrelated stimuli (Fig. 2E).

We have shown that RNNs solved the matching task by eavesdropping on the solution of the prediction task. We have purposefully used a small and simple network, and have optimized the parameters that give rise to eavesdropping (Fig. 2). This now enables us to pinpoint why learning the prediction task catalyzed learning of the matching task.

2.3 Matching task eavesdrops on the stimulus memory of the prediction task

Dual task networks gradually enhanced their neural representation of the S_1 stimulus over the course of training (Fig. 3A). First, the prediction task induced a memory of S_1 until the S_2 presentation (epoch 8, Fig. 3A), which is subsequently strengthened to facilitate the matching computation (Fig. 3B). We define the memory of stimulus S_1 at time t as the decoding accuracy of S_1 of the differentiated neural activity at that time (see Fig. 3A), evaluated across trials using logistic regression (see Methods). Fig. 3C shows how the memory of S_1 dynamically evolved for all three network types after training. Matching single task networks (orange) failed to converge, as they did not create a memory of S_1 beyond the time when S_1 was presented to the networks. However, prediction-only networks (red) created a memory representation of S_1 and maintained this at perfect accuracy until the S_2 presentation, as expected (because the optimal solution would use S_1 information to predict S_2). Dual task networks (blue) also maintained a perfect memory of S_1 after convergence that slightly extends the duration of prediction-only networks, so that S_1 and S_2 could be compared to perform the matching task.

The prediction task and matching task thus share the same latent feature: S_1 memory. The prediction task was solved by creating a memory of S_1, on which dual-task networks could subsequently eavesdrop to solve the matching task (Fig. 3D). Conversely, prediction single task networks that are trained on uncorrelated data ($P(\alpha = \beta) = 0.5$) do not require a memory of S_1 to achieve optimal performance, because S_1 is no longer predictive of S_2. Indeed we found that no S_1 memory was created in this case (Fig. 3F, red), which subsequently failed to bootstrap the matching task for the dual task learning networks (Fig. 3G, H). Hence, only networks trained on predictable stimuli created a memory of S_1 when solving the prediction task, and this facilitated learning of the matching task.

We next investigated how the neural representation of S_1 (as shown in Fig. 3A) dynamically evolved during a trial, which is known to be variable network property (Orhan & Ma, 2019). Intriguingly, we found that dual task networks could embed several different solutions to achieve optimal performance, depending on the stability of the neural representation of S_1. When the subpopulation of neurons that encoded S_1 changed over the course of a trial, this gave rise to a cross-temporal anti-correlation, while an unchanged representation induced a cross-temporally correlated code (Fig. 3I). Finally, a mix of these neuron types decorrelated the memory code. In other words, the cross-temporal correlation of the S_1 representation depended on how variable the identity of its neurons was over time. We quantified

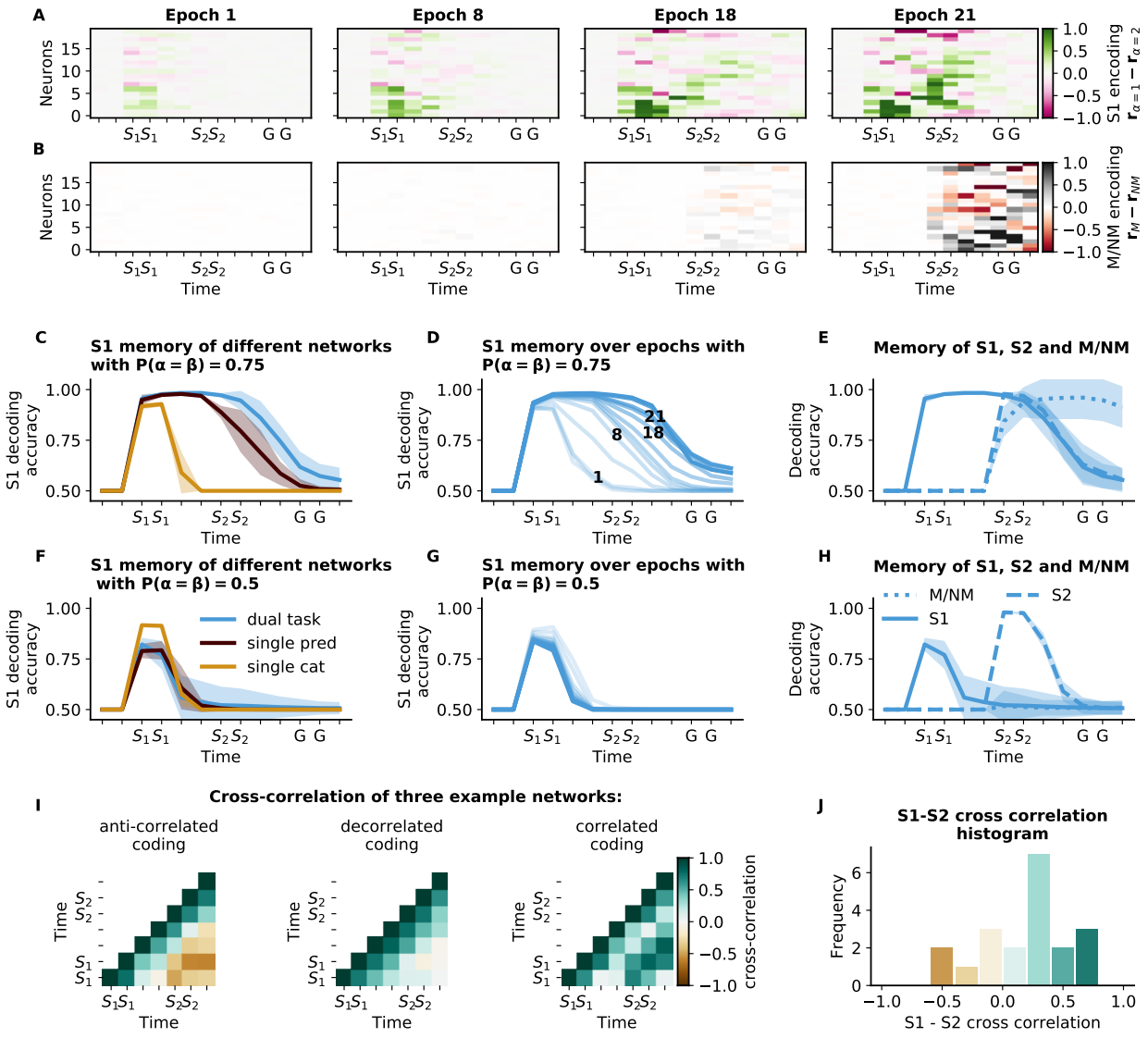

Figure 3: **Matching task eavesdrops on the stimulus memory of the prediction task A)** The neural representation of S_1 is shown for an example network for 4 different epochs during training. Each raster plots shows the differentiated activity between $\alpha = 1$ and $\alpha = 2$ trials of all neurons (clipped between -1 and 1 for clarity). **B)** Similarly, the neural representation of the Match/Non-Match (M/NM) trial type is shown at the same 4 epochs. **C)** The memory of S_1 was determined per time point by decoding the stimulus identity from the network activity. Panels C-H show the average $\pm$ std across 20 simulations. **D)** Dual task networks gradually learned to enhance the memory of S_1 (different epochs are shown from light to dark blue). The 4 epochs of panels A and B are indicated in the figure. **E)** Dual task networks memorize S_1 until the S_2 stimulus is presented, after which they represent the Match/Non-Match identity of the trial. **F, G, H)** Equivalent figures for networks that were trained with uncorrelated stimuli. The prediction task no longer induces a S_1 memory (red), because it is not predictive of S_2 anymore. It subsequently fails to bootstrap the matching task in the dual task networks (blue), because the S_1 memory is lacking. **I)** The cross-correlation of the differentiated activity (as in panel A) is shown for 3 example networks that show anti-correlated, decorrelated and correlated coding respectively (quantified by the S_1-S_2 cross-correlation). **J)** The distribution of S_1-S_2 cross-correlation values of the 20 network simulations (as determined by the average of the 4 cross-temporal data points with S_1 vs S_2 from panel I).

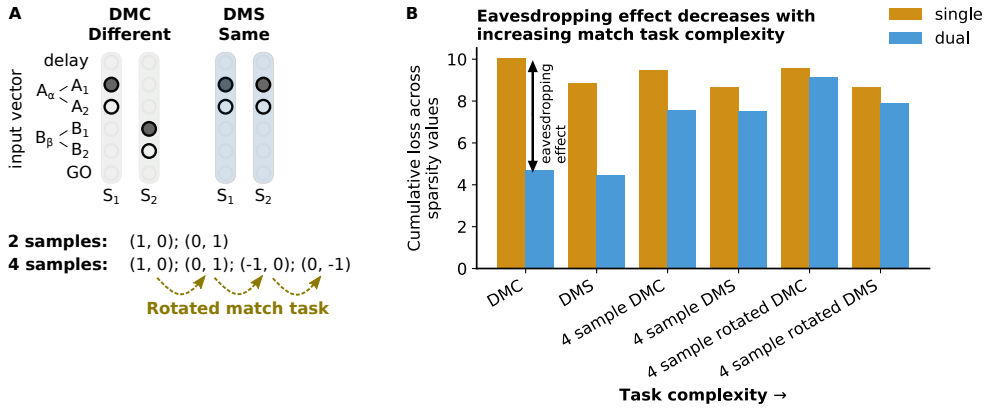

Figure 4: **Increasing matching task complexity reduces eavesdropping effect A)** We investigated the benefit of eavesdropping for 6 different tasks (including the task of previous figures). Each task is a combination of the following 3 conditions: **Input domain**: stimuli S_1 and S_2 either used the same input domain (A_α and A_β, i.e., delayed-match-to-sample (DMS) task) or different domains (A_α and B_β, i.e., delayed-match-to-category (DMC) task). **Number of samples**: each stimulus could either be chosen from 2 samples ($A, B \in \{(1,0),(0,1)\}$) or 4 samples ($A, B \in \{(\cos\theta, \sin\theta); \theta \in \{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\}\}$). Because these stimuli can also take negative values, a different output non-linearity (tanh instead of softmax) and loss function (mean least squares instead of cross-entropy) was used for the prediction task. (We verified that the 2-sample prediction task also converged when optimized with this output nonlinearity and loss function.) **Type of matching task**: For the 4-sample condition, we considered a regular matching task ($A_\alpha = B_\beta$ in the case of different input domains), and a rotated matching task ($A_\alpha = B_{\beta+\frac{\pi}{2}}$). **B)** The effect of eavesdropping was quantified for the 6 different matching tasks, by summing the sparsity-dependent loss of the matching task of Fig. S2 (i.e., the orange and blue lines, experiment was repeated for all tasks, with 20 networks per sparsity value). As the matching task complexity grew, eavesdropping became less effective because the dual-task networks became worse at solving the task.

this variability across different network simulations, which differed only in initial conditions, illustrating the multitude of solutions that networks could use (Fig. 3J). This highlights that eavesdropping was not specific to a particular neural representation of the S_1 stimulus, as each of these solutions maintained the S_1 memory equally well.

2.4 Increasing matching task complexity reduces eavesdropping effect

Eavesdropping allowed RNNs to learn the matching task by utilizing memories from solving the prediction task. Consequently, eavesdropping failed when stimuli were not predictable (Fig. 3F-H), i.e., when S_1 was not predictive of S_2 such that the prediction task was no longer sufficiently informative of the matching task. Next we asked if we could similarly identify the limits of when eavesdropping is an efficient learning strategy by increasing the difficulty of the matching task instead.

We changed the matching task by considering 6 different combinations of three possible conditions: input domain, number of samples and matching task type (Fig. 4A). We then quantified the eavesdropping effect across sparsity values (as in Fig. 2C and Fig. S2) and summed over data points to obtain a single metric. In short, we found that as the matching task complexity increased, eavesdropping gradually became less effective because the dual-task networks could no longer solve the matching tasks. This decrease in eavesdropping efficiency was not caused by a failure of the networks to solve the prediction task, as networks still solved the prediction task at all task complexities and sparsity values (except for the two strongest sparsity values that lead to $\sim 0\%$ nonzero weights), see Fig. S3. Task performance was similar for both input domains (match-to-sample or match-to-category tasks), but decreased when more stimuli or a rotated-match task were introduced (Fig. 4B). Hence, as the difficulty of the matching task increases, eavesdropping on the solution of the prediction task becomes less useful for learning the matching task.

2.5 Simulated annealing enables RNNs to learn the matching task with uncorrelated stimuli

In the previous sections we have shown that eavesdropping only worked with predictable stimuli, because only then a memory of S_1 was created. To prove that the stimulus S_2 predictability - and the thereby induced S_1 memory -

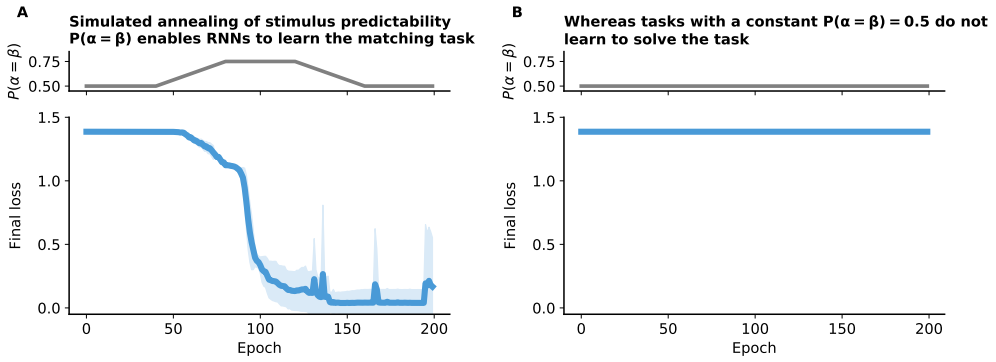

Figure 5: **Simulated annealing enables RNNs to learn the matching task with uncorrelated stimuli A)** Dual task networks are able to learn the matching task with uncorrelated stimuli, by temporarily increasing and decreasing the stimulus S_2 predictability $P(\alpha = \beta)$ from 0.5 to 0.75 and back to 0.5. This simulated annealing of the stimulus predictability causes the prediction task to learn the memory of S_1, which then bootstraps learning of the matching task. Networks are able to maintain their stable solution of the matching task - which does not depend on stimulus S_2 predictability - when the stimuli are uncorrelated again. The end result is a network that has learned the matching task with uncorrelated stimuli. **B)** In contrast, dual task networks are not able to learn the matching task with uncorrelated stimuli without simulated annealing (also see Fig. 2E). Both panels show the mean $\pm$ std of 10 simulated networks per network type.

only helped to initiate the learning process of the matching task, but was not necessary for performing the task after learning converges, we consider the following: we dynamically varied the stimulus S_2 predictability $P(\alpha = \beta)$ from a plateau of 0.5 to a plateau of 0.75 and back again to 0.5. We name this procedure *simulated annealing*, after the commonly-used technique in Monte Carlo sampling (MacKay, 2003, Ch. 30).

We found that dual task networks that engaged in this setting first learned to solve the prediction task using uncorrelated stimuli, and then gradually learned to memorize S_1 for predicting S_2 as stimulus predictability increased. This S_1 memory could then bootstrap learning of the matching task, and networks maintained a stable solution after the stimuli decorrelated again (Fig. 5A). Hence, the matching task could be performed with uncorrelated stimuli, but only by simulated annealing of the stimulus S_2 predictability, as networks with constant $P(\alpha = \beta) = 0.5$ did not converge (Fig. 5B). In summary, a transient period where stimuli are predictable provides the bootstrapping required to learn the demanding matching task, by inducing stimulus memories in the network activity.

3 Discussion

3.1 Summary and Limitations

We have shown that predictive learning acted as a helper task for sparse RNNs to achieve optimal performance in a delayed-match-to-category (DMC) task. This was facilitated by eavesdropping, a multitask learning mechanism whereby the easier prediction task created a persistent neural representation of the stimulus that catalyzed learning of the matching task. Crucially, it was this stimulus representation (or memory) that was necessary for the matching task to initiate learning. Our results show that predictive learning can induce this representation, and was therefore sufficient to initiate learning of the matching task (Fig. 3).

Eavesdropping benefits were specific to certain parameters and task variants. The sparsity optimization showed that eavesdropping is effective for a range of parameter settings that ensure the matching task is neither too difficult (strong sparsity regularization) nor too easy (no sparsity regularization) (Fig. 2). Further, as the matching task complexity increased, networks failed to converge, even with eavesdropping (Fig. 4). Lastly, previous studies have optimized very similar matching tasks (including a variant of the rotated match task) without eavesdropping, using similar RNNs of 100-500 hidden units (Yang et al., 2019; Masse et al., 2019; Orhan & Ma, 2019). Our results show that although smaller networks naturally fail to learn these memory tasks, eavesdropping is able to facilitate success.

Small, interpretable neural networks are crucial for understanding learning mechanisms (Rudin, 2019), and can provide a good approximation of larger biological networks (Barak, 2017). Previous studies have shown that larger networks with more complex architectures can use predictive learning to boost performance in more difficult tasks, such as 3D

navigation (Wayne et al., 2018), but their underlying mechanisms are difficult to dissect. After we confirmed that task performance was equivalent across a range of network sizes (Fig. S1), we adhered to a minimal working example of predictive learning, which enabled us to uncover eavesdropping as its facilitating mechanism (Fig. 3). Notably, the stimulus-prediction and DMC tasks that we considered are typical of controlled experiments performed in animals and humans (Freedman & Assad, 2006; Wan et al., 2020; Mohan et al., 2021; Libby & Buschman, 2021; Zhou et al., 2021), which allows us to use this theoretical study to make experimental predictions that can readily be tested *in vivo*. We will do so at the end of the Discussion section. Hence, crucially, the aim of this paper was not to develop a new Machine Learning heuristic, but rather to use artificial networks to further develop our understanding of biological learning.

3.2 Predictive learning and eavesdropping in natural learning

Eavesdropping is a form of multitask learning where two (or more) tasks that are optimized simultaneously share a latent feature that can only (easily) be learned in one of the tasks, hence catalyzing learning in the other task (Caruana, 1997). Eavesdropping, and multitask learning more generally, is conceptually related to both transfer learning and curriculum learning, which both use 'helper' tasks to improve learning. All these learning strategies have been proposed to occur naturally in animals (Elman, 1993; Ben-David & Schuller, 2003; Bengio et al., 2009; Weiss et al., 2016; Hassabis et al., 2017). However, it remains unclear what the helper task of eavesdropping could be in neural systems.

Simultaneously, predictive learning has been shown to benefit reinforcement learning (RL) agents (Russek et al., 2017; Wayne et al., 2018). In particular, the successor representation - a form of predictive learning (Dayan, 1993; Gershman et al., 2012) - has been able to provide a link between model-free and model-based RL (Russek et al., 2017), which are both known to occur in the brain (Daw et al., 2011).

In neuroscience, brains are thought to continuously engage in predictive learning, often motivated by normative frameworks (Rao & Ballard, 1999; Friston et al., 2016). Our study builds on this, by proposing an additional function of predicting sensory inputs. We have linked eavesdropping and predictive learning, by showing that predictive learning was sufficient to induce eavesdropping in environments where stimuli are correlated in time. Other learning processes that create stimulus representations, such as solving a delayed-response task (Yang et al., 2019; Duncker et al., 2020), could potentially also facilitate eavesdropping. However, in this work we have focused on predictive learning, given its prevalence in neuroscience.

Prediction may be useful as sensory correlations are ubiquitous. Our study suggests that prediction functions as a natural prerequisite for the ability to remember and process sensory experiences over time. Importantly, this does not entail that ongoing predictability is necessary to maintain a memory: In Fig. 5 we show that transient predictability is already enough to bring the system into a new minimum, which remains stable after stimulus predictability disappears. Thus, brief moments during which a complex task can be deconstructed into smaller parts can reveal a path to the solution, after which these components can be disregarded again (also see Knoblich et al. (1999)).

3.3 Experimental predictions

Libby & Buschman (2021) found that mice actively encoded stimulus predictions during a passive learning paradigm (i.e., with no reward structure). Our study suggests that this learning is functionally important: representing stimulus correlations that are not directly relevant for behavior might still be a rewarding long-term strategy. Specifically, predictive learning can act as a module that facilitates learning of complex future task structures.

Taking these experimental findings together with our results, we propose the following hypothesis. Animals that have to solve a complex working memory task, such as DMC, should benefit from prior passive exposure to correlated sequences of the stimuli (that are later used in the DMC task). With no requirement for reward, they should internally attempt to predict these stimuli, thereby creating neural representations of the stimuli that can later accelerate learning. Prior exposure to *uncorrelated* sequences, in contrast, should not benefit learning, as random sequences cannot be predicted.

4 Methods

Software & code availability All data was analyzed and visualized with Python 3.7, using PyTorch as automatic differentiation package, and various other standard libraries for analysis and visualization (Numpy, Scipy, Scikit-Learn, Pandas, Matplotlib, Seaborn, IPython). All experiments and analysis were done on a 16-thread CPU of a desktop computer. All code and trained network simulations are available at: github.com/vdplasthijs/eavesdropping

Terminology Vectors are denoted by bold notation ($\mathbf{x}$) and matrices by capital notation (X). Time is denoted by subscript t, trials are denoted by subscript k.

Tasks We considered two tasks in Figs. 1, 2, 3, 5: first, the prediction task required networks to continuously predict the next upcoming input stimulus, second, the matching task required networks to report whether two stimuli matched at the end of each trial (Fig. 1A).

In Fig. 4 we considered the task described above, and 5 additional variants, by changing 1) the input domain, 2) the number of possible stimulus samples and 3) a direct match or rotated match. These are further described in the caption of Fig. 4.

Synthetic data Each trial consists of a sequence of 2 stimuli S_1 and S_2 and a GO cue, interleaved by delay periods (Fig. 1B). Stimuli are one-hot vectors, with $S_1 = A_\alpha \in \{A_1, A_2\}$ and $S_2 = B_\beta \in \{B_1, B_2\}$ (see Fig. 1B). Hence, the total trial sequence is:

$$\text{trial sequence} = (0, 0, A_\alpha, A_\alpha, 0, 0, B_\beta, B_\beta, 0, 0, \text{GO}, \text{GO}, 0, 0) \tag{1}$$

where each element of Eq. 1 is a one-hot vector. Networks are either trained on *predictable* sequences ($P(\alpha = \beta) = 0.75$), or *unpredictable* (uncorrelated) sequences ($P(\alpha = \beta) = 0.5$). The duration of all stimuli, GO and the delay periods is 2 time points. This ensures that networks that are trained on predictable sequences will encode both stimuli for optimal performance: A_α information is used to predict the first B_β with 75% accuracy, and then the first B_β is used to predict the second B_β with 100% accuracy.

The network input $\mathbf{x}$ is defined by the one-hot vectors of Eq. 1 (Fig. 1B) with added white noise (to prevent networks from overfitting) with standard deviation $\sigma_x = 0.15$ (we found numerically that the results were robust to variations in σ_x). For each network, we randomly generated 1000 trials, which are stratified-split into an 80/20 train/test ratio of trials. All results presented were evaluated on test data only.

Recurrent neural network training We used standard recurrent neural networks (RNNs), consisting of 1 hidden layer of $N = 20$ neurons that all receive input from 6 input neurons and project to 8 output neurons (Fig. 1A). The (rate) activity of neurons $\mathbf{r}_t$ is defined by :

$$\mathbf{r}_t = \tanh\left(U \cdot \mathbf{x}_t + W \cdot \mathbf{r}_{t-1} + \mathbf{b}_W\right) \tag{2}$$

where $\mathbf{x}_t$ is the input at time t, $\mathbf{b}$ is the learned bias, U is the input weight matrix and W the recurrent weight matrix. At the start of each trial, $\mathbf{r}$ is initialized with $\mathbf{r}_{-1} \sim N(0, 0.1 \cdot I)$.

The 6-element prediction task output vector $\hat{\mathbf{y}}_t^{\text{pred}}$ that estimates the next stimulus is given by a softmax function: $\hat{\mathbf{y}}_t^{\text{pred}} = \text{softmax}(V \cdot \mathbf{r}_t + \mathbf{b}_V)$ where V is the prediction output weight matrix. Hence, $\sum_i \hat{y}_{t,i}^{\text{pred}} = 1$, and the network predicts the probability of each possible outcome, rather than one deterministic value. Similarly, the 2 output neurons of the matching task $\hat{\mathbf{y}}_t^{\text{match}}$ must estimate $\mathbf{y}_t^{\text{match}} = (1, 0)^T$ on Match (M, $\alpha = \beta$) trials and $(0, 1)^T$ on Non-Match (NM, $\alpha \neq \beta$) trials, and are defined by $\hat{\mathbf{y}}_t^{\text{match}} = \text{softmax}(\text{ReLU}(T \cdot \mathbf{r}_t + \mathbf{b}_T))$ where T is the task output weight matrix. (Two output units instead of one were used to ensure networks actively had to report both possible outcomes, and an additional ReLU non-linearity was introduced to prevent the trivial solution $T_{\text{M}} = -T_{\text{NM}}$.) The loss function $\mathcal{L}_k$ of trial k, used to train the model parameters $\theta = \{U, W, V, T, \mathbf{b}\}$, is the sum of the cross-entropy loss of either one or both tasks - depending on the network type - and a sparsity L1 regularization term:

$$\mathcal{L}_k = \mathcal{L}_k^{\text{pred}} \cdot \mathbb{1}_{\text{pred}} + \mathcal{L}_k^{\text{match}} \cdot \mathbb{1}_{\text{match}} + \lambda \cdot \mathcal{L}_k^{\text{L1}} \tag{3}$$

where λ is the sparsity regularization parameter (optimized in Fig. 2). During the training phase, the loss $\mathcal{L}_k$ is accumulated during each trial k. At the end of each trial, $\mathcal{L}_k$ is backpropagated through time to update $\theta = \{U, W, V, T, \mathbf{b}\}$ using stochastic gradient descent (SGD) with a learning rate set to 0.002.

Quantifying stimulus representation strength The strength of stimulus representations was quantified in Fig. 3C-H using the stimulus decoding accuracy of logistic regression. We trained decoders to decode the stimulus identity (e.g. S_1) at each time point t: $P(\alpha = 1|\mathbf{r}_t) = 1 - P(\alpha = 2|\mathbf{r}_t)$. Decoders were trained across trials using logistic regression with L1 regularization (scikit-learn implementation with a SAGA solver, `max_iter` $= 250$ and `C` $= 0.1$, (Pedregosa et al., 2011)). To ensure that the stimulus predictability does not bias the decoders, we trained decoders with a new set of 800/200 train/test trials with uncorrelated stimuli $P(\alpha = \beta) = 0.5$. Decoding accuracy is subsequently defined as the mean probability of correctly decoding the stimulus identity.

AUTHOR CONTRIBUTIONS

All authors conceived the project and contributed to writing and editing of the manuscript. TLvdP performed all numerical analyses and wrote the manuscript draft. TPV and SGM supervised the project.

ACKNOWLEDGMENTS

The authors would like to thank members of the Vogels lab and Manohar lab, as well as Adam Packer, Andrew Saxe, Stefano Sarao Mannelli and Jacob Bakermans for fruitful discussions and comments on earlier versions of the manuscript.

TLvdP was supported by funding from the Biotechnology and Biological Sciences Research Council (BBSRC) [grant number BB/M011224/1]. TPV was supported by an ERC Consolidator Grant (SYNAPSEEK). SGM was funded by a MRC Clinician Scientist Fellowship MR/P00878X and Leverhulme Grant RPG-2018-310.

APPENDIX

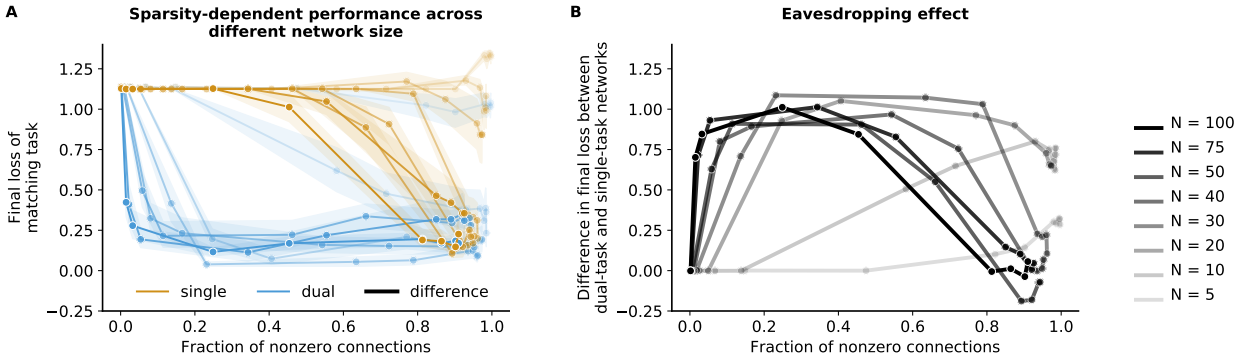

Figure S1: **Eavesdropping effect dependency on network size.** The eavesdropping effect as in Fig. 2C was quantified for a range of network sizes N (i.e., number of hidden units) from 5 to 100. The single task and dual task match losses are shown in panel **A**, and their differences in panel **B**. Network size is indicated by transparency (see legend on the right). Very small networks ($N = 5$) cannot solve the task, while larger networks generally require eavesdropping for optimal performance, especially when sparsity is constrained. The average $\pm$ 95% confidence interval of 50 simulations per data point is shown.

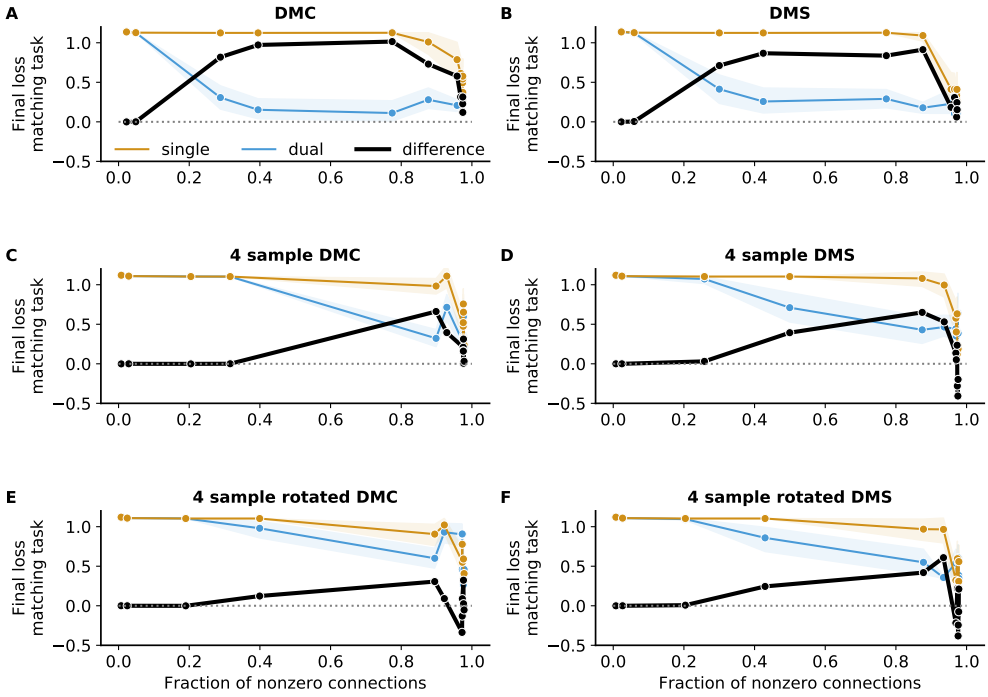

Figure S2: **Detailed breakdown of Fig. 4B.** Each panel is as in Fig. 2C, conducted for each of the 6 tasks as defined in Fig. 4. NB: For each task the same range of sparsity regularization parameter λ values were evaluated, which could result in slight variations in number of nonzero connections (hence the variation of the x-axis positions of the data points across panels). The optimal loss is 0 for all tasks.

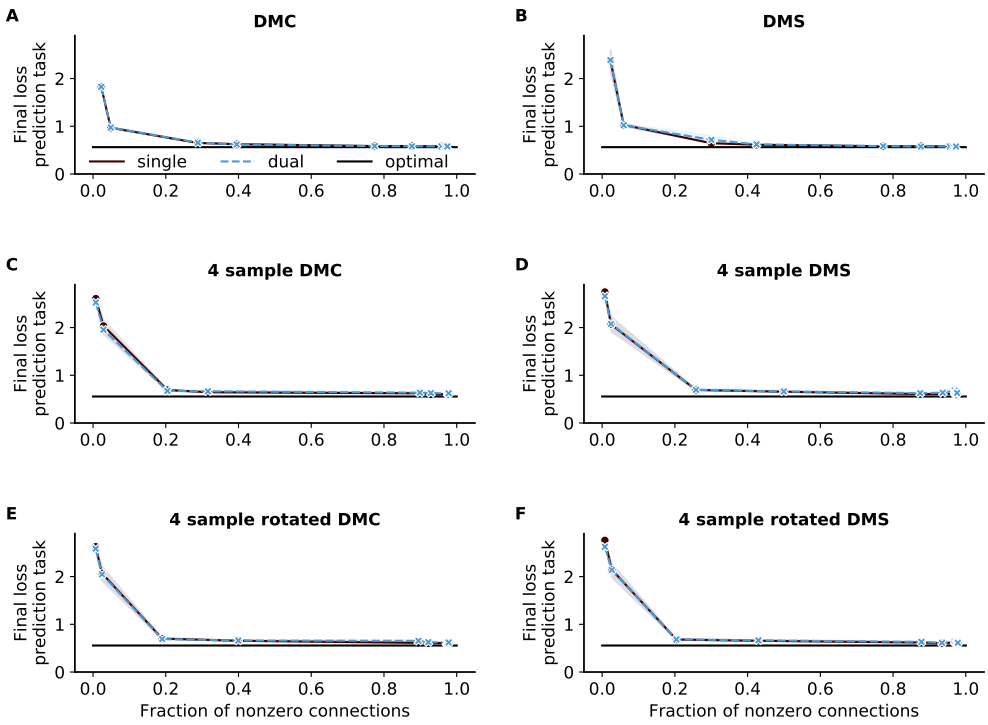

Figure S3: **The prediction task is solved for all tasks.** Each panel shows the prediction task loss of both the dual task networks and single (prediction) task networks, for each of the 6 tasks as defined in Fig. 4. Only networks with severe sparsity regularization, such that almost all connections were 0, failed to learn the task.

REFERENCES

William H Alexander and Joshua W Brown. Frontal cortex function as derived from hierarchical predictive coding. *Scientific reports*, 8(1):1–11, 2018.

Moshe Bar. The proactive brain: memory for predictions. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1521):1235–1243, 2009.

Omri Barak. Recurrent neural networks as versatile tools of neuroscience research. *Current opinion in neurobiology*, 46:1–6, 2017.

Peter Bell and Steve Renals. Regularization of context-dependent deep neural networks with context-independent multi-task training. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4290–4294. IEEE, 2015.

Shai Ben-David and Reba Schuller. Exploiting task relatedness for multiple task learning. In *Learning theory and kernel machines*, pp. 567–580. Springer, 2003.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Nathaniel D Daw, Samuel J Gershman, Ben Seymour, Peter Dayan, and Raymond J Dolan. Model-based influences on humans' choices and striatal prediction errors. *Neuron*, 69(6):1204–1215, 2011.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

Lea Duncker, Laura Driscoll, Krishna V Shenoy, Maneesh Sahani, and David Sussillo. Organizing recurrent network dynamics by task-computation to enable continual learning. *Advances in neural information processing systems*, 33:14387–14397, 2020.

Jeffrey L Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1): 71–99, 1993.

David J Freedman and John A Assad. Experience-dependent representation of visual categories in parietal cortex. *Nature*, 443(7107):85–88, 2006.

Karl Friston, Thomas FitzGerald, Francesco Rigoli, Philipp Schwartenbeck, Giovanni Pezzulo, et al. Active inference and learning. *Neuroscience & Biobehavioral Reviews*, 68:862–879, 2016.

Samuel J Gershman, Christopher D Moore, Michael T Todd, Kenneth A Norman, and Per B Sederberg. The successor representation and temporal context. *Neural Computation*, 24(6):1553–1568, 2012.

Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258, 2017.

Günther Knoblich, Stellan Ohlsson, Hilde Haider, and Detlef Rhenius. Constraint relaxation and chunk decomposition in insight problem solving. *Journal of Experimental Psychology: Learning, memory, and cognition*, 25(6):1534, 1999.

Alexandra Libby and Timothy J Buschman. Rotational dynamics reduce interference between sensory and memory representations. *Nature Neuroscience*, pp. 1–12, 2021.

Ruonan Liu, Boyuan Yang, and Alexander G Hauptmann. Simultaneous bearing fault recognition and remaining useful life prediction using joint-loss convolutional neural network. *IEEE Transactions on Industrial Informatics*, 16(1):87–96, 2020.

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Nicolas Y Masse, Guangyu R Yang, H Francis Song, Xiao-Jing Wang, and David J Freedman. Circuit mechanisms for the maintenance and manipulation of information in working memory. *Nature neuroscience*, 22(7):1159–1167, 2019.

Krithika Mohan, Ou Zhu, and David J Freedman. Interaction between neuronal encoding and population dynamics during categorization task switching in parietal cortex. *Neuron*, 109(4):700–712, 2021.

Yukie Nagai. Predictive learning: its key role in early cognitive development. *Philosophical Transactions of the Royal Society B*, 374(1771):20180030, 2019.

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

A Emin Orhan and Wei Ji Ma. A diverse range of factors affect the nature of neural representations underlying short-term memory. *Nature neuroscience*, 22(2):275–283, 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79–87, 1999.

Stefano Recanatesi, Matthew Farrell, Guillaume Lajoie, Sophie Deneve, Mattia Rigotti, and Eric Shea-Brown. Predictive learning as a network mechanism for extracting low-dimensional latent space representations. *Nature communications*, 12(1):1–13, 2021.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

Evan M Russek, Ida Momennejad, Matthew M Botvinick, Samuel J Gershman, and Nathaniel D Daw. Predictive representations can link model-based reinforcement learning to model-free mechanisms. *PLoS computational biology*, 13(9):e1005768, 2017.

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

Quan Wan, Ying Cai, Jason Samaha, and Bradley R Postle. Tracking stimulus representation across a 2-back visual working memory task. *Royal Society open science*, 7(8):190228, 2020.

Greg Wayne, Chia-Chun Hung, David Amos, Mehdi Mirza, Arun Ahuja, Agnieszka Grabska-Barwinska, Jack Rae, Piotr Mirowski, Joel Z Leibo, Adam Santoro, et al. Unsupervised predictive memory in a goal-directed agent. *arXiv preprint arXiv:1803.10760*, 2018.

Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1): 1–40, 2016.

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

Guangyu Robert Yang, Madhura R Joglekar, H Francis Song, William T Newsome, and Xiao-Jing Wang. Task representations in neural networks trained to perform many cognitive tasks. *Nature neuroscience*, 22(2):297–306, 2019.

Jianfei Yu and Jing Jiang. Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. In *EMNLP 2016: Proceedings of the Conference on Empirical Methods in Natural Language Processing: Austin, Texas, November 1-5*, pp. 236–246. Association for Computational Linguistics, 2016.

Jingfeng Zhou, Chunying Jia, Marlian Montesinos-Cartagena, Matthew PH Gardner, Wenhui Zong, and Geoffrey Schoenbaum. Evolving schema representations in orbitofrontal ensembles during learning. *Nature*, 590(7847): 606–611, 2021.