PREDICTIVE LEARNING IN ENERGY-BASED MODELS WITH ATTRACTOR STRUCTURES

Anonymous authors
Paper under double-blind review

ABSTRACT

Predicting future observations from actions is central to both latent world models in machine learning and internal models in neuroscience. However, many successful world models rely on backpropagation through time and variational inference, making their connection to biologically plausible neural computation unclear. We propose an energy-based recurrent state-space model for action-conditioned predictive learning. The model combines a hierarchical predictive-coding-like energy model for high-dimensional observations with a continuous attractor neural network that serves as a structured memory state. This formulation supports sampling-based inference, local synaptic updates, and memory-based imagination without backpropagating gradients through recurrent memory dynamics. We derive a predictive cross-entropy upper bound optimized by prior-state samples, leading to a learning-before-inference objective that differs from the posterior-state objectives commonly used in variational RSSMs and predictive coding models. Experiments on visual prediction tasks involving eye movements, navigation-like motion, head turning, and passive temporal changes show that the model can generate stable predictions from partial observations and generalize to unseen environments. Comparisons with recurrent, predictive-coding, and machine-learning baselines suggest that attractor-structured memory provides a useful inductive bias for biologically inspired predictive world modeling.

1 INTRODUCTION

Intelligent agents need to predict the sensory consequences of their actions. Such action-conditioned prediction is central to the notion of an internal model in neuroscience Knill & Pouget (2004); Friston & Price (2001) and to the world-model framework in machine learning Schmidhuber (1990); Ha & Schmidhuber (2018); Hafner et al. (2023). Recent machine-learning systems have demonstrated that predicting future observations in a latent space can support representation learning, planning, and imagination. However, most high-performing latent world models rely on architectures and learning algorithms, such as backpropagation through time (BPTT), that are not directly aligned with biologically plausible neural computation.

A parallel line of work in neuroscience and cognitive modeling has studied predictive coding, active inference, energy-based models, and attractor networks as candidate mechanisms for perception, inference, and memory Rao & Ballard (1999); Friston (2010); Ackley et al. (1985); Amari (1977). Predictive coding and energy-based formulations naturally express inference as neural dynamics and learning as local error-driven updates. Continuous attractor neural networks (CANNs), on the other hand, provide a canonical model for structured memory and continuous latent variables in neural systems Ben-Yishai et al. (1995); Wu et al. (2008). Yet it remains challenging to combine these ingredients into a single predictive model that can process high-dimensional observations, update memory from action-observation sequences, and generate future observations through imagination.

In this paper, we propose a biologically inspired computational framework for action-conditioned predictive learning. Our framework is an energy-based recurrent state-space model (RSSM), in which a memory state summarizes past observations and actions, a prior over latent neural activity is generated from this memory, and future observations are predicted through a hierarchical energy-based decoder. The model is implemented with a hierarchical predictive-coding-like network for high-dimensional sensory variables and a CANN for structured recurrent memory.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

A key difference from standard variational RSSMs is the order of learning and inference. In conventional inference-then-learning approaches, the model first infers a posterior latent state after observing the current input and then optimizes a reconstruction or evidence lower bound (ELBO)-like objective under this posterior. In contrast, our model optimizes prediction using samples from the prior latent state generated by memory before observing the current input. This learning-before-inference objective directly trains the memory-conditioned prior to predict the upcoming observation. After learning, posterior inference is performed to update the current latent state and memory. This separation leads to a biologically plausible online process: predict from memory, update synapses using prediction error, infer the current latent state, and then update memory.

We evaluate the model on visual prediction tasks involving eye movements, movement in virtual environments, movement and head turning in real environments, and passive temporal changes. These tasks test whether the model can predict high-dimensional observations after actions, whether attractor-structured memory can support long-range imagination, and whether the learned memory state generalizes beyond memorized trajectories. In particular, predictions on unseen images and unseen sequences provide an important test of generalization, distinguishing our model from attractor-memory models that primarily retrieve stored episodes without generalizing to new observations.

Our contributions are summarized as follows:

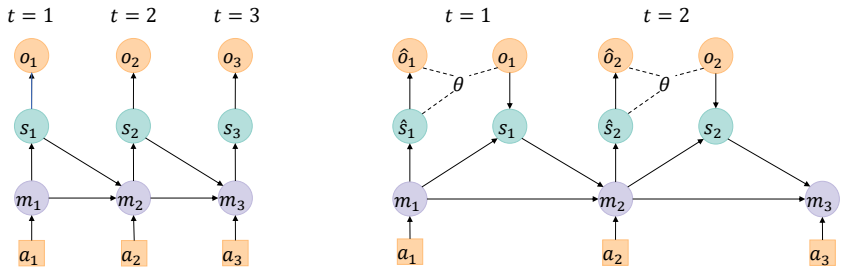
- We formulate action-conditioned predictive learning as an energy-based recurrent state-space model, in which prediction, learning, inference, and memory update are expressed within a unified probabilistic framework.
- We instantiate this framework with a hierarchical predictive-coding-like energy model and a continuous attractor memory, yielding local synaptic updates and sampling-based inference without BPTT through recurrent memory dynamics.
- We introduce a learning-before-inference objective based on prior-state prediction, and clarify its relation to posterior-state objectives such as ELBO-based variational learning.
- We show across several visual prediction tasks that the model supports stable prediction, imagination from partial observations, and generalization to unseen environments, while remaining competitive with recurrent and predictive-coding baselines under biologically motivated constraints.

2 Related Work

Latent world models. World models learn compact latent representations for predicting future observations after actions Schmidhuber (1990); Ha & Schmidhuber (2018). Recurrent state-space models (RSSMs), such as PlaNet and Dreamer, encode high-dimensional observations into latent states and use recurrent dynamics for prediction and planning Hafner et al. (2019; 2020; 2023). Later works have replaced recurrent dynamics with Transformers, S4 models, or other sequence models to improve long-horizon prediction Chen et al. (2022); Samsami et al. (2024). These models provide strong engineering solutions for latent prediction, but they are usually trained with backpropagation and BPTT. Our work follows the RSSM perspective but asks a different question: how can an RSSM-like predictive model be implemented with local neural dynamics and structured memory?

Predictive coding and active inference. Predictive coding networks (PCNs) model perception and learning through hierarchical prediction errors Rao & Ballard (1999); Whittington & Bogacz (2017). Many PCN models focus on static inputs or supervised learning Salvatori et al. (2021; 2023); Millidge et al. (2022), while recent extensions introduce temporal dynamics and action-conditioned prediction Ororbia & Mali (2023); Tang et al. (2023). Active inference provides another influential framework in which perception and action are formulated as free-energy minimization under a generative model Friston (2010); Friston et al. (2017); Smith et al. (2022). These approaches are closely related to posterior-state learning, where inference is performed before parameter updates. In contrast, our framework emphasizes a learning-before-inference objective: the model first predicts from the memory-conditioned prior, updates parameters according to the predictive error, and then performs posterior inference for memory update.

Energy-based neural models. Energy-based models (EBMs) provide a natural language for describing inference as energy minimization or sampling and learning as local changes in energy landscapes Ackley et al. (1985). Their connection to neural sampling and Hebbian learning has made



(a) Generative model

(b) Prediction learning & inference

Figure 1: (a) The directed graphical model of the generative model. Taking $t = 2$ as an example, s_2 follows the prior, o_2 follows the likelihood function, and m_3 follows the transition probability. (b) Taking $t = 2$ as an example, the brain initially generates a prediction $\hat{s}_2$ for the neural activity, followed by producing a prediction $\hat{o}_2$ for the observation. Then the network parameters are updated, as indicated by the dashed lines, and the posterior for the current time step is obtained. At last, the memory is updated based on action a_3 and the sample s_2 following the posterior.

them attractive for biologically plausible computation Hoyer & Hyvärinen (2002); Haefner et al. (2016); Orbán et al. (2016); Echeveste et al. (2020). Hierarchical EBMs can also be interpreted as predictive-coding-like models when their energies are written as layer-wise prediction errors Dong & Wu (2023). Our model extends this perspective to recurrent predictive learning by coupling a hierarchical EBM with a memory-conditioned latent prior.

Attractor memory and cognitive maps. Attractor networks have long been used to model memory, spatial representation, and cognitive maps Amari (1972); Hopfield (1982); Samsonovich & McNaughton (1997). Continuous attractor neural networks are especially suitable for representing continuous variables such as position, orientation, and other low-dimensional latent states Amari (1977); Ben-Yishai et al. (1995); Wu et al. (2008). Recent attractor-memory models can retrieve stored episodes but often focus on memory reconstruction rather than generalization to unseen observations Chandra et al. (2025). In contrast, our CANN is embedded inside a predictive encoder-decoder framework. This allows the model not only to store action-observation history, but also to generate predictions for unseen images and unseen sequences from partial observations.

3 Energy-based recurrent state space model

We formulate action-conditioned prediction as an energy-based recurrent state-space model. The goal is not to claim that the brain literally implements the exact model below, but to provide a candidate computational framework in which prediction, learning, inference, and memory update can be realized by neural dynamics and local synaptic updates. The detailed neural implementation is presented in Section 4.

Problem setup. Consider an agent that takes action a_t and then receives observation o_t. Because the full physical state of the environment is not directly observed, the current observation alone is generally insufficient for predicting future observations. The agent therefore maintains a memory state m_t that summarizes the past action-observation history $K_t = \{o_{<t}, a_{\leq t}\}$. The predictive task is to model the distribution of the upcoming observation o_t conditioned on the memory m_t.

Generative model. At each time step, the model first samples a latent state s_t from a memory-conditioned prior and then generates an observation from a conditional likelihood:

$$p_\theta(o_t, s_t|m_t) = p_\theta(o_t|s_t)p(s_t|m_t). \tag{1}$$

Here s_t denotes latent neural activity, and m_t denotes the memory state before observing o_t. The prior $p(s_t|m_t)$ represents the model's prediction of the latent state from memory, while $p_\theta(o_t|s_t)$ maps latent activity to sensory observations. After the observation is processed, the memory is updated according to

$$m_{t+1} \sim p(m_{t+1}|m_t, s_t, a_{t+1}). \tag{2}$$

In this paper, the memory transition is implemented by deterministic neural dynamics, corresponding to a Dirac transition distribution. This makes the model an energy-based recurrent state-space model.

Prediction. Before observing o_t, the model predicts the upcoming observation by sampling from the marginal distribution

$$p_\theta(o_t|m_t) = \int p_\theta(o_t|s_t)p(s_t|m_t)\, ds_t. \tag{3}$$

Operationally, this is implemented by first sampling a prior latent state $\hat{s}_t \sim p(s_t|m_t)$, and then generating a predicted observation $\hat{o}_t \sim p_\theta(o_t|\hat{s}_t)$. Thus, the quality of prediction depends on whether the memory-conditioned prior $p(s_t|m_t)$ provides latent samples that are informative about the upcoming observation. (Figure 1b).

Learning. After the true observation $o_t \sim p_{\text{true}}(o_t)$ is received, the model updates its parameters to improve future prediction. The ideal predictive objective is the cross-entropy between the true observation distribution and the model prediction:

$$\mathcal{H} = -\mathbb{E}_{o_t\sim p_{\text{true}}(o_t)} \log p_\theta(o_t|m_t). \tag{4}$$

Since

$$p_\theta(o_t|m_t) = \mathbb{E}_{s_t\sim p(s_t|m_t)} p_\theta(o_t|s_t),$$

Jensen's inequality gives

$$\begin{aligned} \mathcal{H} &= -\mathbb{E}_{o_t\sim p_{\text{true}}(o_t)} \log \mathbb{E}_{s_t\sim p(s_t|m_t)} p_\theta(o_t|s_t) \\ &\le -\mathbb{E}_{o_t\sim p_{\text{true}}(o_t)} \mathbb{E}_{\hat{s}_t\sim p(s_t|m_t)} \log p_\theta(o_t|\hat{s}_t) =: \mathcal{L}. \end{aligned} \tag{5}$$

Thus, $\mathcal{L}$ is a predictive cross-entropy upper bound. Importantly, the expectation is taken over prior latent samples $\hat{s}_t \sim p(s_t|m_t)$ generated before observing o_t. Optimizing this bound therefore directly trains the memory-conditioned prior to produce latent states that predict the upcoming observation. We write the conditional likelihood in energy-based form:

$$p_\theta(o_t|s_t) = \frac{\exp[-E_\theta(o_t, s_t)]}{Z_\theta(s_t)}. \tag{6}$$

In the general EBM case, learning involves both positive and negative phases. In the Gaussian implementation used in Section 4, the conditional energy is quadratic and the precision matrix is fixed, making the partition term independent of θ. This yields a local prediction-error learning rule. For the general energy-based form, the gradient of $\mathcal{L}$ can be written as

$$\nabla_\theta \mathcal{L} = \mathbb{E}_{\hat{s}_t\sim p(s_t|m_t)} \Big[\mathbb{E}_{o_t\sim p_{\text{true}}(o_t)} \nabla_\theta E_\theta(o_t, \hat{s}_t) - \mathbb{E}_{\hat{o}_t\sim p_\theta(o_t|\hat{s}_t)} \nabla_\theta E_\theta(\hat{o}_t, \hat{s}_t) \Big], \tag{7}$$

where the first term lowers the energy of observed samples and the second term raises the energy of model-generated samples. In our Gaussian hierarchical implementation, this expression reduces to layer-wise local prediction-error updates.

Prediction-before-inference learning. The objective in Eq. (5) differs from the posterior-state objectives commonly used in variational RSSMs and predictive-coding models. In a standard inference-then-learning paradigm, one first infers a posterior latent state after observing o_t: $s_t \sim p_\theta(s_t|m_t, o_t)$, and then optimizes a reconstruction or ELBO-like objective of the form

$$\mathbb{E}_{s_t\sim p_\theta(s_t|m_t,o_t)} \log p_\theta(o_t|s_t). \tag{8}$$

This posterior expectation is closely related to variational learning, where the latent state used for learning already contains information from the current observation. In contrast, our learning-before-inference objective uses the prior latent state:

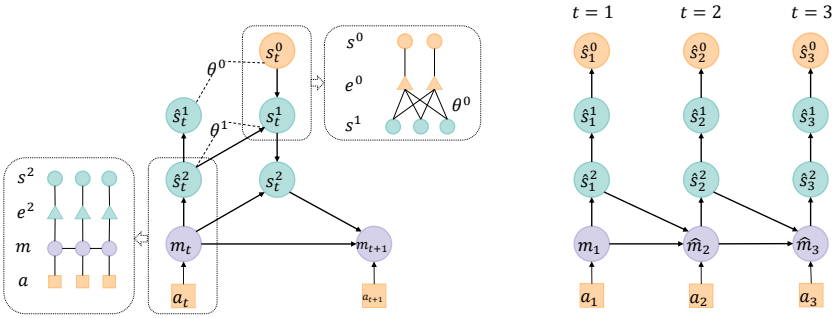
$$\mathbb{E}_{\hat{s}_t\sim p(s_t|m_t)} \log p_\theta(o_t|\hat{s}_t), \tag{9}$$

where $\hat{s}_t$ is sampled from memory before observing o_t. The key difference is therefore the distribution used to generate the latent sample:

$$\text{learning-before-inference:} \quad \hat{s}_t \sim p(s_t|m_t), \tag{10}$$

$$\text{inference-then-learning:} \quad s_t \sim p_\theta(s_t|m_t, o_t). \tag{11}$$

The former directly evaluates whether the memory-conditioned prior can predict the upcoming observation, whereas the latter evaluates whether a posterior state can reconstruct an already observed input. This distinction is important for action-conditioned prediction: the model should improve the



(a) Prediction learning & inference (b) Imagination

Figure 2: (a) In the hierarchical structure, the activity of neurons in the upper layer is the observation for the neurons in the lower layer. In the case of $L = 2$, s_t^0 is the observation of s_t^1, and s_t^1 is the observation of s_t^2. The bottom-left box illustrates the connection of the memory-representing CANN with action neurons and its link to s^2 through e^2. The top-right box shows the connections between s^0 and s^1 through e^0. (b) Based on the current memory m_1, we imagine the observations we would receive at each time step after taking a series of actions.

latent states generated from memory before the observation arrives. After this predictive learning step, posterior inference is still performed, but its role is to update the current latent state and memory rather than to define the learning target.

Inference & memory update. After updating the likelihood function $p_\theta(o_t|s_t)$, the brain also needs to update the neural representation s_t and the memory representation m_t. Specifically, the new distribution of s_t becomes,

$$p_{\text{post}} = \underset{q}{\text{argmax}} \left\{ \mathbb{E}_q \ln p_\theta(o_t|s_t) - D_{\text{KL}} \left[q || p(s_t|m_t) \right] \right\} \propto p_\theta(o_t|s_t) p(s_t|m_t). \tag{12}$$

This new distribution reflects that, on one hand, under this distribution, we can better predict the true observation, i.e., maximizing $\mathbb{E}_q \log p_\theta(o_t|s_t)$. This target also implies enabling s_t to contain as much information from o_t as possible (refer to the derivation in Appendix B). Meanwhile, we aim to minimize variation in the neural representation, i.e., ensuring that the new distribution remains close to the previous. To strike a balance between these two objectives, the new distribution takes the form of the p_{post}. The brain can use the sampling-based approach to obtain the distribution of p_{post}, such as the Langevin dynamic,

$$\tau_s \frac{\mathrm{d}s}{\mathrm{d}t} = \nabla_s \log p_\theta(o_t|s_t) + \nabla_s \log p(s_t|m_t) + \sqrt{2\tau_s}\xi, \tag{13}$$

where ξ is Gaussian white noise and τ_s is the time constant. At last, we use the samples of the distribution $p_{\text{post}}(s_t)$ to update the memory according to the generative model,

$$m_{t+1} \sim p(m_{t+1}|m_t, s_t, a_{t+1}), \quad s_t \sim p_{\text{post}}(s_t). \tag{14}$$

4 A HIERARCHICAL NEURAL NETWORK MODEL

We now instantiate the energy-based RSSM with a hierarchical neural network. The hierarchy implements the conditional likelihood over high-dimensional observations, while a continuous attractor neural network implements the recurrent memory state. This implementation is biologically inspired in the sense that inference is performed by neural dynamics and parameter updates are local to pre-synaptic activity, post-synaptic activity, and prediction-error neurons.

A hierarchical generative model. Let $s_t^0 \in \mathbb{R}^{n_0}$ be the observation variable. There are L layers of neurons representing the latent variables $s_t^{1:L} = \{s_t^1, s_t^2, ..., s_t^L\}$, $s_t^l \in \mathbb{R}^{n_l}$. The joint distribution is a Markov chain,

$$p_\theta(s_t^{0:L}|m_t) := p(s_t^L|m_t) \prod_{l=0}^{L-1} p_\theta(s_t^l|s_t^{l+1}). \tag{15}$$

To avoid calculating the derivation of the partition function in the likelihood function, we utilize the Gaussian distribution, whose partition function is constant,

$$p_\theta(s_t^l|s_t^{l+1}) := \mathcal{N}(s_t^l;\ \theta^l f(s_t^{l+1}),\ (\Lambda^l)^{-1}), \tag{16}$$

where $f(\cdot)$ is the element-wise activation function and Λ^l is the inverse of the covariance matrix called precision matrix. $\theta_l \in \mathbb{R}^{n_l \times n_{l+1}}$ are parameters which determine the connectivity structure of the network.

The memory network. Attractor neural networks have been widely used as canonical models for elucidating the memory process in the neural system Amari (1972); Hopfield (1982). Among these, CANNs excel in capturing continuous variables, as the underlying state of a temporal sequence is typically continuous. Therefore, we use the activity of CANN neurons at time t as the memory m_t. The CANN, through its recurrent connections, forms a series of continuous attractors. This sequence of attractors constitutes a stable low-dimensional manifold, serving as the memory space. In spatially-related tasks, it is also referred to as a cognitive map O'keefe & Nadel (1979); Samsonovich & McNaughton (1997). The CANN receives inputs from actions and from the last layer's latent neurons for prediction and memory update (see Appendix C for the CANN dynamics). When the CANN receives inputs from action a_t and neurons s^L_{t-1}, it generates activity m_t, which gives rise the neuronal activity in the last layer according to,

$$p(s^L_t|m_t) := \mathcal{N}(s^L_t;\ m_t,\ (\Lambda^L)^{-1}). \tag{17}$$

Prediction. At time t, the neural network first generates the prediction samples $\hat{s}^{0:L}_t$ from layer L to 0 according to,

$$\hat{s}^L_t \sim p(s^L_t|m_t), \quad \hat{s}^l_t \sim p(s^l_t|\hat{s}^{l+1}_t). \tag{18}$$

We use the Langevin dynamic to generate predictions,

$$\tau_s \frac{\mathrm{d}s^l}{\mathrm{d}t} = \nabla_{s^l} p(s^l|\hat{s}^{l+1}_t) + \sqrt{2\tau_s}\xi = -\Lambda^l \hat{e}^l_t + \sqrt{2\tau_s}\xi, \tag{19}$$

where $\hat{e}^l_t = s^l_t - \theta^l f(\hat{s}^{l+1}_t)$ and $\hat{e}^L_t = s^L_t - m_t$ are the value represented by error neurons. We adopt the idea of predictive coding networks Rao & Ballard (1999); Whittington & Bogacz (2017) by introducing error neurons, to satisfy Hebb's rule during learning. The connectivity diagram of neurons is depicted by the dashed box in Figure 2a.

Learning & inference. After the model receives the observation $s^0_t \sim p_{\text{true}}(s^0_t)$, for s^1 represented by neurons in layer one, the likelihood function is $p(s^0_t|s^1_t)$ and the prior distribution is $p(s^1_t|\hat{s}^2_t)$. Thus, the prediction bound $\mathcal{L}^0_t$ can be written as,

$$\begin{aligned}\mathcal{L}^0_t &:= -\mathbb{E}_{s^0_t \sim p_{\text{true}}(s^0_t)} \mathbb{E}_{s^1_t \sim p(s^1_t|\hat{s}^2_t)} \ln p(s^0_t|s^1_t), \\ &= \frac{1}{2}\left(\hat{e}^0_t\right)^T \Lambda^0 \hat{e}^0_t + C^0,\end{aligned} \tag{20}$$

where C^0 is the constant. Then, synaptic parameters θ^0 are updated to minimize $\mathcal{L}^0_t$ using gradient descent,

$$\tau_\theta \frac{\mathrm{d}\theta^0}{\mathrm{d}t} = -\nabla_{\theta^0} \mathcal{L}^0_t = \Lambda^0 \hat{e}^0_t f(\hat{s}^1_t)^T. \tag{21}$$

This shows that the synaptic changes are determined solely by local neurons, adhering to Hebb's rule. Then neurons in layer one keeps a balance between the likelihood and the prior by inferring the posterior $p^1_{\text{post}} \propto p(s^0_t|s^1_t)p(s^1_t|\hat{s}^2_t)$ through Langevin dynamic,

$$\begin{aligned}\tau_s \frac{\mathrm{d}s^1_t}{\mathrm{d}t} &= \nabla_{s^1_t} \ln p(s^0_t|s^1_t) + \nabla_{s^1_t} p(s^1_t|\hat{s}^2_t) + \sqrt{2\tau_s}\xi, \\ &= f'(s^1_t) \odot (\theta^0)^T \Lambda^0 e^0_t - \Lambda^1 \hat{e}^1_t + \sqrt{2\tau_s}\xi,\end{aligned} \tag{22}$$

where $e^0_t = s^0_t - \theta^0 f(s^1_t)$ and $\odot$ is the element-wise product. Then the sample s^1_t following the posterior will be used to minimize $\mathcal{L}^1_t$ and obtain sample $s^2_t \sim p^2_{\text{post}}$. Each layer will repeat this process and propagate information downward until the last layer (see second for-loop in Algorithm 1). For random vector s^l in layer l. The prediction bound $\mathcal{L}^l_t$ is calculated as,

$$\begin{aligned}\mathcal{L}^l_t &:= -\mathbb{E}_{s^{l-1}_t \sim p^{l-1}_{\text{post}}} \mathbb{E}_{s^l_t \sim p(s^l_t|\hat{s}^{l+1}_t)} \ln p(s^{l-1}_t|s^l_t), \\ &= \frac{1}{2}\left(\hat{e}^l_t\right)^T \Lambda^l \hat{e}^l_t + C^l,\end{aligned} \tag{23}$$

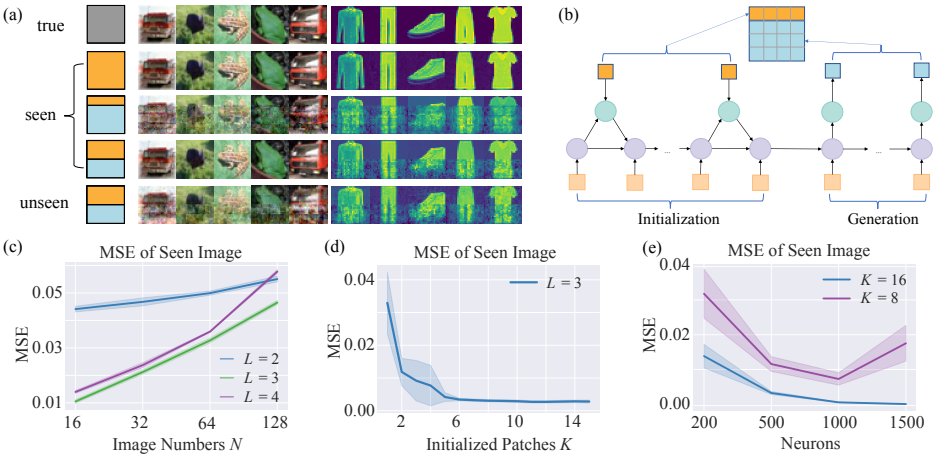


Figure 3: Experiments on modeling eye movement. (a) Generation results of the entire image through initialized memory. The orange and blue areas in the first column indicate the patches used for initialization and those that need to be predicted. 'Seen' refers to images that were used during training, while 'unseen' refers to images that were not used during training (here, both are the same images because different models were used). (b) Testing phase: the first step is initialization, similar to the training process, involves executing actions, observing the environment, inferring the latent state, and finally updating the memory. However, unlike training, network weights are not updated in this step. The second step involves executing random actions and predicting observations (c) Model tested by memory initialized with $K = 4$ patches. (d) Model trained on $N = 32$ images. When initialized with $K = 6$ patches , the whole image can be almost completely reconstructed. (e) The x-axis neuron numbers represent the number of neurons in each layer, with a total of $L = 3$ layers in the network.

where C^l is the constant. The posterior p^l_{post} of variables in layer l is calculated as,

$$p^l_{\text{post}} \propto p(s^{l-1}_t | s^l_t) p(s^l_t | \hat{s}^{l+1}_t). \tag{24}$$

After the variables in layer L converge to their posterior, they serve as inputs to the CANN. Meanwhile, the action a_{t+1} at time $t+1$ is also fed into the CANN. The CANN, following its dynamics, reaches a new steady state with the neuron activity denoted as m_{t+1}. Subsequently, the brain utilizes m_{t+1} as the memory to initiate a new round of prediction. Algorithm 1 illustrates the entire process of neural implementation.

Imagination. After our model learns to predict the next observation, it acquires an intrinsic representation of the dynamics of the external environment. If we want to know the outcome of a certain action, there is no need to actually perform the action; instead, we can rely on the model to predict the observation we would receive, called imagination. We can continually make predictions in the latent space, incorporating them into memory, and forecast observations after a sequence of actions (Figure 2b).

5 Experiment

We evaluate our model by selecting four types of action in different environments, including eye movement, motion in a virtual environment, motion and head-turning in a real environment, and static observation while the external world varies. To simulate the high-dimensional inputs received by the brain, all observations in our study are exclusively chosen to be visual inputs. We refer to the appendix for hyper parameters (Appendix E).

Eye movement refers to changing the direction of the eyeballs to obtain different visual inputs. It stands as the most frequent actions performed by humans, helping us gather as much visual information as possible. To avoid dizziness caused by rapid eye movement, neurons in the posterior parietal cortex encode stimuli that will be seen after planned eye movements Cui & Andersen (2011); Kuang et al. (2016). Additionally, experiments Seung (1996) suggest that neurons in the medial vestibular nucleus form a CANN to record eye direction.

We utilized the CIFAR-10 and Fashion-MNIST datasets to simulate the environments observed by the model. Each image in the dataset is divided into 4×4 patches, with each patch serving as an input for a single observation to mimic the human receptive field. After each eye movement, the next observation becomes the corresponding patch.

During the learning phase (Figure 2a), we randomly generate a sequence of eye movement, and change the complete image every once in a while. We employed $N = 16, 32, 64, 128$ images for each model, resulting in a total of $N_{\text{tol}} = 16 \times N$ possible observations. Rows 2-4 of Figure 3a demonstrate the generation results for images encountered during training, while the 5th row illustrates the generation results for unseen images. Figure 3c shows the mean squared error (MSE) between the prediction and the ground truth for different network structures (total number of neurons is the same, L varies), which decreases with training epochs. To increase training efficiency, we used a batch size of 128, and roughly, the effectiveness of one epoch can be considered as the average over 128 time steps. Figure 3d depicts the decreases of loss $\mathcal{L}_t^l$ for each layer in the $L = 3$ model across training epochs. Here, the phenomenon of gradient vanishing is observed, and we plan to address this by introducing skip connections to deepen the network. Figure 3e displays the impact of network capacity on performance. When initializing memory with $K = 16$ patches, a higher number of neurons corresponds to improved model performance. However, when initializing with $K = 8$ patches, an excessive number of neurons increases the initialization space, posing a challenge and resulting in a decline in model performance.

During the testing phase (Figure 3b), we start by randomly initializing the memory. We select K patches from an image and perform random eye movement on these patches for prediction and inference without altering the network weights. After $2K$ steps, the obtained memory becomes the initialized memory. We then use this memory to envision each patch, generating the whole image. Although the CIFAR-10 dataset has higher dimensions than Fashion-MNIST, we found that the model performs better on CIFAR-10. This may be attributed to the fact that CIFAR-10, as a natural image dataset, exhibits higher correlations between patches, which is more favorable for the model's predictive capabilities. We also compared our model's test results on CIFAR-10 with the commonly used TransDreamer model (TDM) Chen et al. (2022) for similar tasks in machine learning methods. The results show that our model achieved better performance with the same number of parameters (Table 1). In appendix D, we provide more details about the training process and clearer generated results.

Motion and head-turning alter our spatial position and the orientation of our head, respectively. Experimental findings reveal neurons encoding position and head orientation in the brain exhibit structures akin to CANNs Wills et al. (2005); Kim et al. (2017). Notably, place cells, particularly located in the hippocampus, are believed to be closely associated with spatial cognition and memory functions Moser et al. (2015).

In our experiments, we employed an agent moving in four directions within the Deeplab map Beattie et al. (2016), constructing a dataset for a virtual environment using observed images and action sequences. Additionally, we utilized the Google Street View Static API to capture images by continuously moving and rotating the viewpoint, creating a dataset for a real environment. Each environment was associated with datasets of varying sizes. The first and third rows of Figure 4a display observation sequences for the two datasets, while the corresponding action sequences are illustrated in the two rows of labels. During the training phase, each environment underwent continuous training using models with distinct structures. After a maximum of 100 epochs, all models achieved convergence. In the testing phase, for each model, we used a random sequences from the training phase to initialize memory through continuous prediction inference. Subsequently, we executed imagination to predict observations for the entire environment. The second and fourth rows

K	4		8		16	
N	Ours	TDM	Ours	TDM	Ours	TDM
16	0.0907	0.1678	0.0687	0.1321	0.0388	0.0532
32	0.0834	0.1431	0.0629	0.1272	0.0336	0.0512
64	0.0802	0.1281	0.0612	0.1130	0.0304	0.0482
128	0.0770	0.1179	0.0606	0.0911	0.0287	0.0471

Table 1: MSE of predictions calculated on modeling eye movement. We compare our model with transdreamer Chen et al. (2022) for different Image numbers and initialized patches.

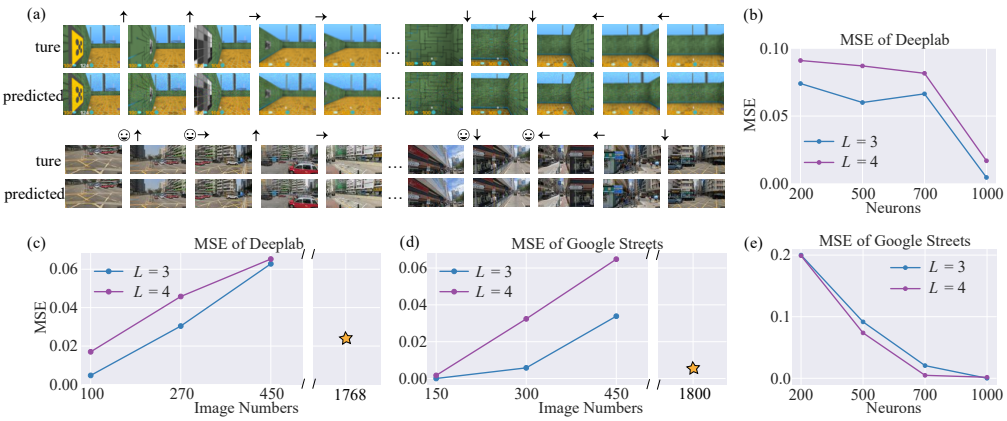


Figure 4: Experiments on modeling motion and head turning. (a) The prediction results. Arrow labels indicate the direction of movement actions, while the combination of the head and arrow illustrates the direction of head-turning actions. (b)(e) The x-axis neuron numbers represent the number of neurons in each layer, with training using 150 images. (c)(d) In both figures, the total number of neurons in the model is 3k, and orange stars mark the results when the number of neurons is 10k.

of Figure 4a showcase the results of predictions. Figures 4b and 4e respectively demonstrate the predictive capabilities of the models for the environments post-training. As evident, an increase in the number of neurons correlates with enhanced predictive capabilities. Figures 4c and 4d illustrate that, with the same network size, larger datasets result in diminished model performance.

Static observation while the environment changes. When we take no action, remaining stationary, the external world can change continuously. For example, when watching a video, our observations constantly evolve. In such cases, we also need to predict future observations. We conducted evaluations using the MNIST-rot dataset and TaxiBJ dataset. The MNIST-rot dataset consists of sequences with 20 frames each, while the TaxiBJ dataset sequences contain 8 frames. During training, we utilized $N = 128$ sequences. In testing, we use sequences that were not seen during training. For the MNIST-rot dataset, we initialized memory with the first 10 frames and then reproduced the entire sequence. For the TaxiBJ dataset, memory was initialized with the first 4 frames, and the entire sequence was reproduced from the beginning (Figure 6 in appendix). We achieved better results than tPCN on the MNIST-rot dataset (Table 3). And we compared our performance on the TaxiBJ dataset with BP-based machine learning models, achieving comparable results (Table 2).

6 Discussion

We consider that the brain employs an EBM as an intrinsic generative model to predict the next observation after action. We utilize a hierarchical neural network to implement this process and incorporate a CANN as memory to compress past experiences. As a biologically plausible neural network, our model succeeds in various environments with different actions. This provides insight into how the brain builds an internal model capturing the dynamics of the environment. Moreover, our model achieves performances on par with machine learning methods, indicating that our framework has the potential for further development.

Unlike previous machine learning works Hafner et al. (2020) or predictive coding network approaches Tang et al. (2023), our framework differs in that we first perform learning and then inference. In contrast to previous approaches that conduct learning after inference, this order in our framework leads to a distinct objective function. Our objective is expressed as $\mathbb{E}_{p(s|m)} \log p(o|s)$ (see Eq.(5)), whereas previous works often use $\mathbb{E}_{p_{\text{post}}} \log p(o|s)$. We experimented with the latter objective as well, but it resulted in poorer and less robust performance in our model. In fact, our approach of learning before inference is closer to the autoregressive models used in LLMs.

In the current model, when computing the gradient of the objective function with respect to the model parameters, we ignore the influence of parameters on memory (see Eq.(7)). This can be understood as a form of Truncated BPTT. Nevertheless, in experiments related to movement, we found that this does not affect our long-distance predictions.

AI USE STATEMENT

Generative AI tools assisted in organizing the authors' model description, drafting manuscript text, preparing the LaTeX project. The authors reviewed all AI-assisted text and figures for accuracy and consistency with the implemented method. The authors take responsibility for the final content of this work, including all AI-assisted text and figures.

REPRODUCIBILITY STATEMENT

The paper specifies the probabilistic objective, inference and learning dynamics, memory update, datasets, baselines, and evaluation procedure. The appendices provide derivations, supplementary results, pseudocode, compute resources, and the full hyperparameter table needed to reproduce the reported experiments. The submission is anonymized for double-blind review.

IMPACT STATEMENT

This work develops a biologically inspired predictive-learning model. It does not involve human subjects or introduce sensitive datasets. Potential downstream uses inherit the usual risks of predictive visual models, including unreliable predictions when deployed outside the evaluated settings; such applications should therefore validate accuracy and robustness before use.

REFERENCES

David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

S-I Amari. Learning patterns and pattern sequences by self-organizing nets of threshold elements. *IEEE Transactions on computers*, 100(11):1197–1206, 1972.

Shun-ichi Amari. Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological cybernetics*, 27(2):77–87, 1977.

Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.

R Ben-Yishai, R Lev Bar-Or, and H Sompolinsky. Theory of orientation tuning in visual cortex. *Proceedings of the National Academy of Sciences*, 92(9):3844–3848, 1995.

Sarthak Chandra, Sugandha Sharma, Rishidev Chaudhuri, and Ila Fiete. Episodic and associative memory from spatial scaffolds in the hippocampus. *Nature*, 638(8051):739–751, 2025.

Chang Chen, Yi-Fu Wu, Jaesik Yoon, and Sungjin Ahn. Transdreamer: Reinforcement learning with transformer world models. *arXiv preprint arXiv:2202.09481*, 2022.

He Cui and Richard A Andersen. Different representations of potential and selected motor plans by distinct parietal areas. *Journal of Neuroscience*, 31(49):18130–18136, 2011.

Xingsi Dong and Si Wu. Neural sampling in hierarchical exponential-family energy-based models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=SWU8YLlFVH.

Rodrigo Echeveste, Laurence Aitchison, Guillaume Hennequin, and Máté Lengyel. Cortical-like dynamics in recurrent circuits optimized for sampling-based probabilistic inference. *Nature neuroscience*, 23(9):1138–1149, 2020.

Karl Friston. The free-energy principle: a unified brain theory? *Nature reviews neuroscience*, 11(2): 127–138, 2010.

Karl Friston, Thomas FitzGerald, Francesco Rigoli, Philipp Schwartenbeck, and Giovanni Pezzulo. Active inference: a process theory. *Neural computation*, 29(1):1–49, 2017.

Karl J Friston and Cathy J Price. Dynamic representations and generative models of brain function. *Brain research bulletin*, 54(3):275–285, 2001.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Ralf M Haefner, Pietro Berkes, and József Fiser. Perceptual decision-making as probabilistic inference by neural sampling. *Neuron*, 90(3):649–660, 2016.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

Patrik Hoyer and Aapo Hyvärinen. Interpreting neural response variability as monte carlo sampling of the posterior. *Advances in neural information processing systems*, 15, 2002.

Nal Kalchbrenner, Aäron Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Video pixel networks. In *International Conference on Machine Learning*, pp. 1771–1779. PMLR, 2017.

Sung Soo Kim, Hervé Rouault, Shaul Druckmann, and Vivek Jayaraman. Ring attractor dynamics in the drosophila central brain. *Science*, 356(6340):849–853, 2017.

David C Knill and Alexandre Pouget. The bayesian brain: the role of uncertainty in neural coding and computation. *TRENDS in Neurosciences*, 27(12):712–719, 2004.

Shenbing Kuang, Pierre Morel, and Alexander Gail. Planning movements in visual and physical space in monkey posterior parietal cortex. *Cerebral cortex*, 26(2):731–747, 2016.

Beren Millidge, Tommaso Salvatori, Yuhang Song, Rafal Bogacz, and Thomas Lukasiewicz. Predictive coding: Towards a future of deep learning beyond backpropagation? *arXiv preprint arXiv:2202.09467*, 2022.

May-Britt Moser, David C Rowland, and Edvard I Moser. Place cells, grid cells, and memory. *Cold Spring Harbor perspectives in biology*, 7(2):a021808, 2015.

John O'keefe and Lynn Nadel. Précis of o'keefe & nadel's the hippocampus as a cognitive map. *Behavioral and Brain Sciences*, 2(4):487–494, 1979.

Marc Oliu, Javier Selva, and Sergio Escalera. Folded recurrent neural networks for future video prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 716–731, 2018.

Gergő Orbán, Pietro Berkes, József Fiser, and Máté Lengyel. Neural variability and sampling-based probabilistic representations in the visual cortex. *Neuron*, 92(2):530–543, 2016.

Alexander Ororbia and Ankur Mali. Active predictive coding: Brain-inspired reinforcement learning for sparse reward robotic control problems. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3015–3021. IEEE, 2023.

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.

Tommaso Salvatori, Yuhang Song, Yujian Hong, Lei Sha, Simon Frieder, Zhenghua Xu, Rafal Bogacz, and Thomas Lukasiewicz. Associative memories via predictive coding. *Advances in neural information processing systems*, 34:3874–3886, 2021.

Tommaso Salvatori, Ankur Mali, Christopher L Buckley, Thomas Lukasiewicz, Rajesh PN Rao, Karl Friston, and Alexander Ororbia. Brain-inspired computational intelligence via predictive coding. *arXiv preprint arXiv:2308.07870*, 2023.

Mohammad Reza Samsami, Artem Zholus, Janarthanan Rajendran, and Sarath Chandar. Mastering memory tasks with world models. *arXiv preprint arXiv:2403.04253*, 2024.

Alexei Samsonovich and Bruce L McNaughton. Path integration and cognitive mapping in a continuous attractor neural network model. *The Journal of Neuroscience*, 17(15):5900–5920, 1997.

Jiirgen Schmidhuber. Making the world differentiable: On using self-supervised fully recurrent n eu al networks for dynamic reinforcement learning and planning in non-stationary environments, 1990.

H Sebastian Seung. How the brain keeps the eyes still. *Proceedings of the National Academy of Sciences*, 93(23):13339–13344, 1996.

Ryan Smith, Karl J Friston, and Christopher J Whyte. A step-by-step tutorial on active inference and its application to empirical data. *Journal of mathematical psychology*, 107:102632, 2022.

Mufeng Tang, Helen Barron, and Rafal Bogacz. Sequential memory with temporal predictive coding. *arXiv preprint arXiv:2305.11982*, 2023.

James CR Whittington and Rafal Bogacz. An approximation of the error backpropagation algorithm in a predictive coding network with local hebbian synaptic plasticity. *Neural computation*, 29(5): 1229–1262, 2017.

Tom J Wills, Colin Lever, Francesca Cacucci, Neil Burgess, and John O'Keefe. Attractor dynamics in the hippocampal representation of the local environment. *Science*, 308(5723):873–876, 2005.

Si Wu, Kosuke Hamaguchi, and Shun-ichi Amari. Dynamics and computation of continuous attractors. *Neural Computation*, 20(4):994–1025, 2008.

Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

A LOWER BOUND DERIVATION 1

Firstly, we can prove that,

$$p(o|s)\log p(o|s) - \log p(o|s), \tag{25}$$
$$= [p(o|s) - 1]\log p(o|s), \tag{26}$$
$$\geq 0 \tag{27}$$

Then, the mutual information between sample $o \sim p_{\text{true}}(o)$ and s under p distribution is calculated as,

$$\mathbb{E}_{o\sim p_{\text{true}}(o)} I(o,s|m) = \mathbb{E}_{o\sim p_{\text{true}}(o)}\mathbb{E}_{s,m\sim p(o,s,m)} \log \frac{p(o,s,m)p(m)}{p(o,m)p(s,m)}, \tag{28}$$
$$= \mathbb{E}_{o\sim p_{\text{true}}(o)}\mathbb{E}_{s,m\sim p(o,s,m)} \log \frac{p(o,s,m)}{p(s,m)} + \mathbb{E}_{o\sim p_{\text{true}}(o)}\mathbb{E}_{m\sim p(m)} \log \frac{p(m)}{p(o,m)}, \tag{29}$$
$$= \mathbb{E}_{o\sim p_{\text{true}}(o)}\mathbb{E}_{s,m\sim p(o,s,m)} \log p(o|s) + \text{Const}, \tag{30}$$
$$\stackrel{+}{=} \mathbb{E}_{o\sim p_{\text{true}}(o)}\mathbb{E}_{m\sim p(m)}\mathbb{E}_{p(s|m)} p(o|s)\log p(o|s), \tag{31}$$
$$\geq \mathbb{E}_{m\sim p(m)}\mathbb{E}_{o\sim p_{\text{true}}(o)}\mathbb{E}_{p(s|m)} \log p(o|s). \tag{32}$$

The relation between $\mathcal{H}$ and $\mathcal{L}$ is written as,

$$\mathcal{L} = \mathcal{H} + \mathbb{E}_{o\sim p_{\text{true}}(o)} D_{\text{KL}}\left[p(s|m)||p_{\text{post}}\right]. \tag{33}$$

B LOWER BOUND DERIVATION 2

The mutual information between o and s under q distribution is calculated as,

$$I(o,s|m) = \mathbb{E}_{o,s,m\sim q(o,s,m)} \log \frac{q(o,s,m)q(m)}{q(o,m)q(s,m)}, \tag{34}$$
$$= \mathbb{E}_{o,s,m\sim q(o,s,m)} \log \frac{q(o,s,m)}{q(s,m)} + \mathbb{E}_{o,m\sim q(o,m)} \log \frac{q(m)}{q(o,m)}, \tag{35}$$
$$= \mathbb{E}_{o,s,m\sim q(o,s,m)} \log q(o|s) + \text{Const}, \tag{36}$$
$$\stackrel{+}{=} \mathbb{E}_{o,s,m\sim q(o,s,m)} \log q(o|s), \tag{37}$$
$$\geq \mathbb{E}_{o,s,m\sim q(o,s,m)} \log q(o|s) - \mathbb{E}_{s\sim q(s,m)} D_{\text{KL}}\left[q(o|s)||p(o|s)\right], \tag{38}$$
$$= \mathbb{E}_{o,m\sim q(o,m)}\mathbb{E}_{s\sim q(s|o,m)} \log p(o|s). \tag{39}$$
$$\tag{40}$$

C THE CANN DYNAMICS

We use $m_t \in \mathbb{R}^{n_L}$ to represent the firing rate of the CANN at time t, and $I_t \in \mathbb{R}^{n_L}$ to represent the total synaptic input to the neurons in CANN. According to the integral firing model, the firing rate can be approximated as,

$$m_t = H(I_t), \tag{41}$$

where $H(\cdot)$ is a nonlinear function. The dynamics of I_t are determined by its own relaxation, recurrent inputs from other neurons, neural adaptation V_t, and external inputs from s_t^L and action neurons, as expressed by the following equation:

$$\tau_I \frac{\mathrm{d}I_t}{\mathrm{d}t} = -I_t + Wm_t - V_t + s_t^L + a_t \tag{42}$$

Here, τ_I represents the synaptic time constant, and W denotes the recurrent neuronal connections. W is a randomly generated low-rank matrix, where its norm is controlled by the hyperparameter $\alpha = \|W\|$. The dynamic of V_t is written as,

$$\tau_V \frac{\mathrm{d}V_t}{\mathrm{d}t} = -V_t + \beta m_t \tag{43}$$

where τ_V is the adaptation time constant and β is a scalar controlling the adaptation strength.

D SUPPLEMENTARY FIGURES

Figure 5a illustrates the learning process of the eye movement experiment. Figure 5b presents clearer generated images, while Figures 5c and 5d depict the changes in MSE and layer losses during the model training process in the eye movement experiment. Figure 6 showcases the experimental results from static observations as the environment changes.

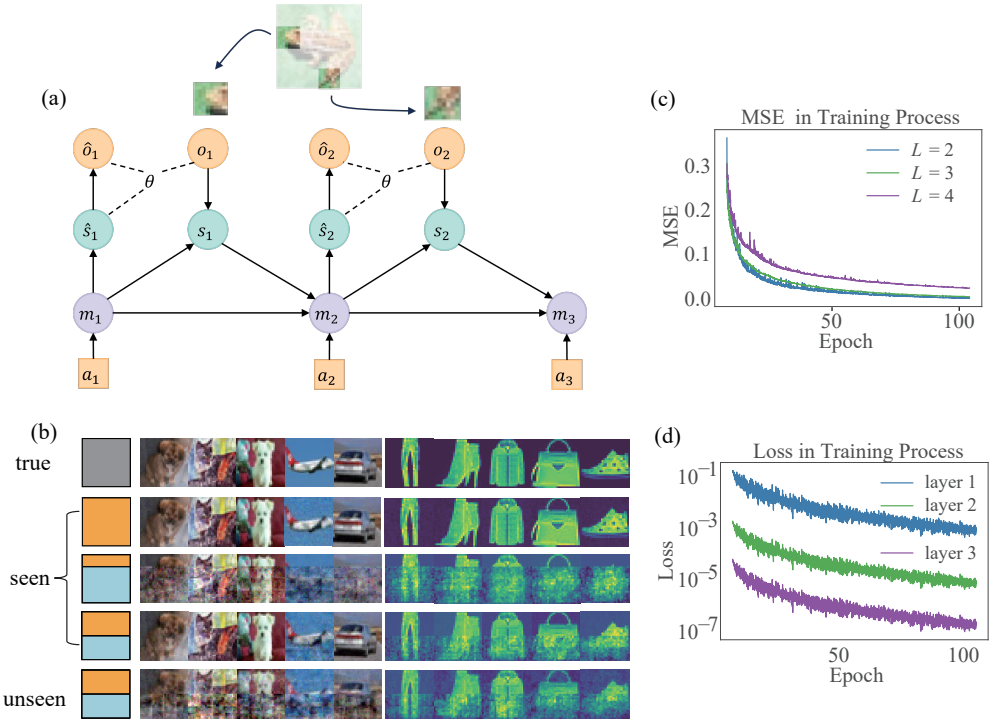


Figure 5: (a) The learning process of the eye movement experiment. (b) Supplement to Figure 3a in the main text. (c) The dataset used here is CIFAR-10, with a total neuron count of $\sum_l n_l = 3$k being consistent across models with different layers L. (d) The loss decreases exponentially as the number of layers increases, with the model trained on $N = 64$ images.

	SEEN		UNSEEN	
SEQLEN	OURS	TPC	OURS	TPC
16	3.44E-8	0.0120	0.0489	0.0545
32	3.52E-7	0.0182	0.0413	0.0451
64	2.65E-5	0.0124	0.0344	0.0347
128	5.10E-4	0.0107	0.0218	0.0232
256	0.0019	0.0105	0.0170	0.0182
512	0.0033	0.0091	0.0112	0.0167
1024	0.0038	0.0096	0.0087	0.0151

Table 2: MSE of predictions calculated on MNIST-rot dataset. We compare our model with tPC Tang et al. (2023) for different sequence length.

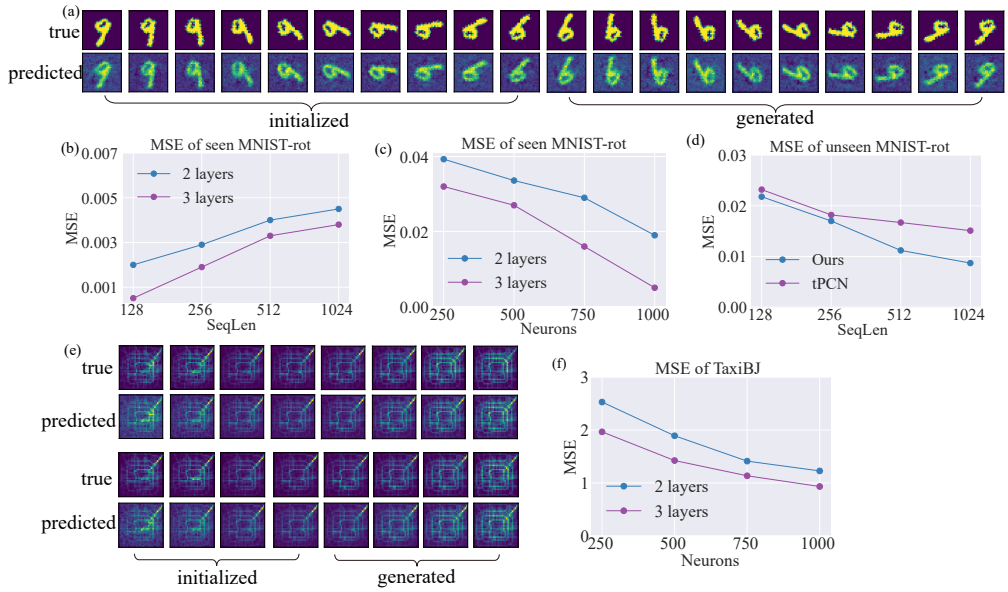


Figure 6: Experiments on sequential data. (a) The 1st row shows the ground truth, and the 2nd row presents the predicted results. We use $K = 10$ frames to initialize memory in the MNIST-rot dataset. (b)(c) Detailed experiments on the MNIST-rot dataset under various parameters. (d) Comparison results with tPCN. (e) Rows 1 and 3 correspond to the ground truth, while Rows 2 and 4 represent the predictions generated by the model. We use $K = 4$ frames for the TaxiBJ dataset. (f) Detailed experiments on the TaxiBJ dataset under different parameters.

MODEL	FRAME 1	FRAME 2	FRAME 3	FRAME 4
ST-RESNET	0.460	0.571	0.670	0.762
VPN	0.427	0.548	0.645	0.721
FRNN	0.331	0.416	0.518	0.619
OURS	0.458	0.514	0.567	0.633

Table 3: MSE of predictions calculated on TaxiBJ dataset. We compare our model with several popular methods in machine learning, including ST-ResNet Zhang et al. (2017), VPN Kalchbrenner et al. (2017), and FRNN Oliu et al. (2018). All compared models take 4 historical traffic flow images as inputs, and predict the next 4 images.

Algorithm 1: Hierarchical neural process

while *still alive* **do**
  `// from time t to t+1`
  Sample $\hat{s}_t^L \sim p(s_t^L|m_t)$;
  for $l \leftarrow L-1$ **to** 0 **do**
    Sample $\hat{s}_t^l \sim p(s_t^l|\hat{s}_t^{l+1})$;
  Receive observation s_t^0;
  for $l \leftarrow 0$ **to** $L-1$ **do**
    Update θ^l by $\frac{\mathrm{d}\theta^l}{\mathrm{d}t} = -\nabla_{\theta^l}\mathcal{L}_t^l$;
    Sample $s_t^{l+1} \sim p_{\text{post}}^{l+1}$ by $\tau_s \frac{\mathrm{d}s_t^{l+1}}{\mathrm{d}t} = \nabla_{s_t^{l+1}} \ln p_{\text{post}}^{l+1} + \sqrt{2\tau_s}\xi$;
  Update $m_{t+1} \sim p(m_{t+1}|m_t, s_t^L, a_{t+1})$.

E HYPER PARAMETERS

Here are the hyperparameters used in the experiment. All programs are run on one NVIDIA RTX A6000, and we use JAX (CUDA 11) to accelerate the programs. For the experiments depicted in our

figures, each one takes 5-20 minutes, with the best performance on the DeepLab and Google Street datasets requiring about 10 hours. The code will be open-sourced after publication.

For all stochastic differential equations, we employ the Euler method for simulation with a step size of $\mathrm{d}t$. Both inference and learning after a single observation were conducted over a total simulation time of T. In other words, for a time step from t to $t+1$, the simulation occurs $T/\mathrm{d}t$ times. The duration of operation for each layer is uniform. All models use the leaky_relu activation function denoted as $f(\cdot)$. The parameter τ_s,τ_I and τ_V are set to 1 in all models.

In Tables 3 and 2, we used the corresponding parameter settings from the original texts. For the TransDreamer in Table 1, the parameter settings are as follows:

- Attention head = 8, Dropout = 0.2, Hidden size = 128, Model size = 64, Layers = 3

DATASET	n_0	L	n_l $(l>0)$	$\mathrm{d}t$	T	$1/\tau_\theta$	α	β	EPOCHS
FASHION-MNIST	7×7	2	1500,1500	0.05	10	0.1	1 0.5 0.1	0	125
		3	1000,1000,1000						
		4	750,750,750,750						
CIFAR-10	$3\times 8\times 8$	2	1500,1500	0.05	10	0.1	1 0.5 0.5 0.1	0	125
		3	1000,1000,1000						
		3	512,256,128						
		4	750,750,750,750						
DEEPMIND LAB	$3\times 80\times 60$	3	1000,1000,1000	0.05	10	0.01	0.5	0	40
		4	750,750,750,750	0.05			0.1		40
		4	4000,2000,2000,2500	0.02			0.1		100
GOOGLE STREET	$3\times 100\times 50$	3	1000,1000,1000	0.05	5	0.1	0.5	0	40
		4	750,750,750,750	0.05	7	0.1	0.1		40
		4	4000,2000,2000,2000	0.1	5	0.05	0.1		140
MNIST-ROT	28×28	3	2000,1000,1000	0.05	10	0.1	0.5	1	100
TAXIBJ	32×32	3	2000,1000,1000	0.05	10	0.1	0.5	1	200

Table 4: Parameters setting for different models