Spiking recurrent neural networks represent task-relevant neural sequences in rule-dependent computation

Xiaohe Xue[1], Michael M. Halassa [2], Zhe S. Chen[3,4,5*]

1 Courant Institute of Mathematical Sciences, New York University, New York, New York, United States of America

2 Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, Massachusetts, United States of America

3 Department of Psychiatry, New York University School of Medicine, New York, New York, United States of America

4 Department of Neuroscience and Physiology, New York University School of Medicine, New York, New York, United States of America

5 Neuroscience Institute, New York University School of Medicine, New York, New York, United States of America

* Corresponding email: zhe.chen@nyulangone.org

ORCID 0000-0002-6483-6056 (Z. S. Chen)

ORCID 0000-0003-1386-0336 (M. M. Halassa)

Abstract

Prefrontal cortical neurons play in important roles in performing rule-dependent tasks and working memory-based decision making. Motivated by experimental data, we develop an excitatory-inhibitory spiking recurrent neural network (SRNN) to perform a rule-dependent two-alternative forced choice (2AFC) task. We imposed several important biological constraints onto the SRNN, and adapted the spike frequency adaptation (SFA) and SuperSpike gradient methods to update the network parameters. These proposed strategies enabled us to train the SRNN efficiently and overcome the vanishing gradient problem during error back propagation through time. The trained SRNN produced rule-specific tuning in single-unit representations, showing rule-dependent population dynamics that strongly resemble experimentally observed data in rodent and monkey. Under varying test conditions, we further manipulated the parameters or configuration in computer simulation setups and investigated the impacts of rule-coding error, delay duration, weight connectivity and sparsity, and excitation/inhibition (E/I) balance on both task performance and neural representations. Overall, our modeling study provides a computational framework to understand neuronal representations at a fine timescale during working memory and cognitive control.

Author Summary

Working memory and decision making are fundamental cognitive functions of the brain, but the circuit mechanisms of these brain functions remain incompletely understood. Neuroscientists have trained animals (rodents or monkeys) to perform various cognitive tasks while simultaneously recording the neural activity from specific neural circuits. To complement the experimental investigations, computational modeling may provide an alternative way to examine the neural representations of neuronal assemblies during task behaviors. Here we develop and train a spiking recurrent neural network (SRNN) consisting of balanced excitatory and inhibitory neurons to perform the rule-dependent working memory tasks Our computer simulations produce qualitatively similar results as the experimental findings. Moreover, the imposed biological constraints on the

trained network provide additional channel to investigate cell type-specific population responses, cortical connectivity and robustness. Our work provides a computational platform to investigate neural representations and dynamics of cortical circuits a fine timescale during complex cognitive tasks.

Introduction

A biological brain is large-scale neuronal network with recurrent connections that performs computations in complex task behaviors. In recent years, recurrent neural networks (RNNs) have been widely used for modeling a wide range of neural circuits, such as the prefrontal cortex (PFC), parietal cortex, and primary motor cortex (M1), in various cognitive and motor tasks [1–8]. Varying types of assumptions have been made in different models: leaky integrate-and-fire (LIF) vs. conductance-based compartment model, rate vs. spiking-based model. Spikes are the fundamental language that brain uses to represent and communicate information. Spiking neural networks (SNNs) are inspired by information processing in the brain, and attempt to mimic the mechanistic operation and computation among spiking neurons [9]. Although SNNs are derived from biologically inspired neurons, neural plasticity or learning for SNNs have been less efficient (i.e., slow convergence) compared to the rate-based models [10–13]. In recent years, many methods have been developed for improving the learning speed of feedforward and recurrent SNNs [14–17].

Working memory and attention control are two fundamental functions in cognition and decision making. The two-alternative forced choice (2AFC) task is a classic experiment in psychophysics in electrophysiology to measure the working memory and cognitive control capacity. In light of recent experiments [18, 19], PFC neurons are found to show rule-specific and timing-specific neuronal responses during the delay (i.e., working memory) period, emerging rule-specific neural sequences (Fig. 1). Sequentially activated neuronal activities ("neural sequences") have been widely observed in neural assemblies in cortical areas during working memory or decision making [18, 20, 21]. In the literature, neural sequences have been generated by RNNs [4, 22]. However, these sequences were trained by supervised learning; it remains unknown whether the rule-specific neural representations or sequences can emerge from a trained RNN. It is

also unclear how these neural representations change as a function of network excitation and inhibition. Inspired by these questions, we developed an excitatory-inhibitory SRNN model that satisfies Dale's principle [7, 23], and we adapted a SuperSpike surrogate gradient learning algorithm [16] and SFA method [24, 25] to train the SRNN. Unlike the rate-based RNN, SRNN can provide millisecond resolution to characterize neuronal tuning.

The contributions of this paper are threefold. First, to alleviate the vanishing gradient problem, we adapted the SuperSpike surrogate gradient algorithm and SFA method to learn excitatory-inhibitory SRNN parameter. These combined computational strategy is key to efficiently train the SRNN for a delayed working memory task. Second, we showed the trained SRNN produced emergent task-relevant neural representations of excitatory and inhibitory neurons, such as neural sequences and oscillatory spiking during the delay period. Our computer model prediction match the published experimental data. Third, we manipulated and quantified the impact of the excitatory-to-inhibitory (E/I) balance, delay duration, and cortical connectivity on task performance and neural representations. Overall, our large-scale SRNN modeling and simulations provide a computational framework to study task representations and dynamics in a wide range of cognitive tasks.

Methods

Rule-dependent working memory

In the 2AFC task, animals were trained to initiate a trial upon receiving broadband white noise. Upon successful initiation, the white noise was immediately replaced by either low-pass or high-pass filtered noise for 100 ms to indicate the rule (i.e., attending to vision vs. audition). This is was followed by a delay period lasting ~ 400 ms before the target stimuli presentation. Animals received a reward if their behavior led to a correct rule-specific response at two specific response ports (Fig. 1A, [18]). In an extension of 2AFC task, animals were trained to respond to four response ports instead of two response ports (known as 4AFC task), whereas the other task phases remained unchanged. Responses were scored as correct or one of three different error types

(executive error, sensory error, or both). All animal experiments were performed according to the guidelines of the US National Institutes of Health and the Institutional Animal Care and Use Committee at the New York University Langone Medical Center.

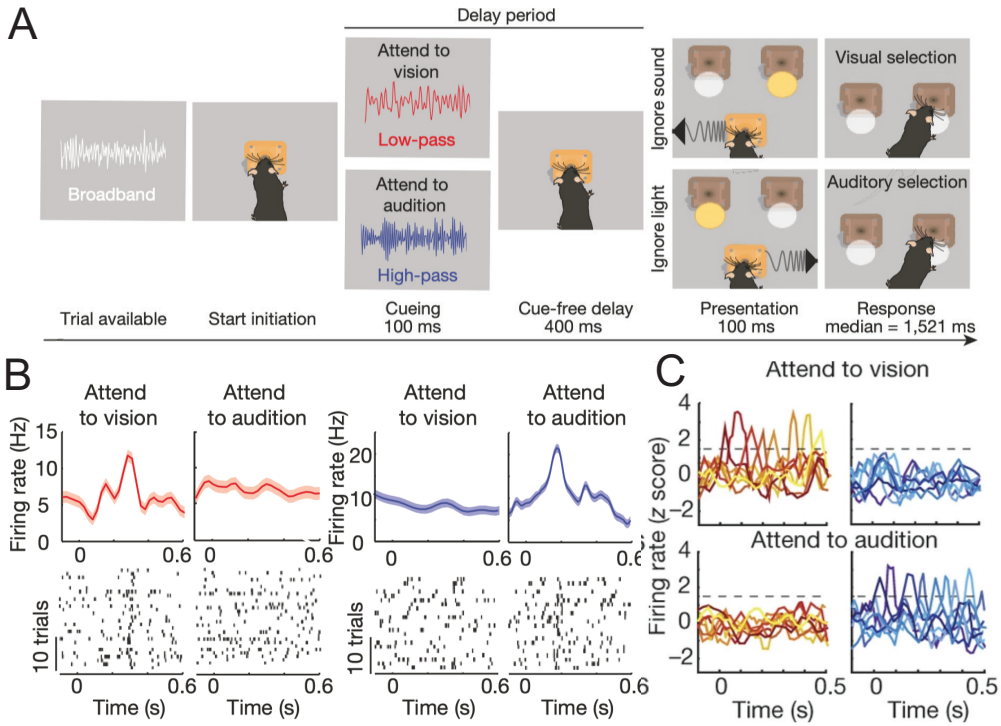

Fig 1. Rule-specific sequential PFC activity emerges during a rodent working memory task. (A) Schematic of 2AFC task. (B) Representative peri-stimulus time histogram (PSTH) and rasters for mouse PFC neurons with specific tuning of attending to vision (rule 1) or attending to audition (rule 2). (C) Examples of tuning peaks from mouse PFC neurons across multiple sessions (Figures reproduced from [18] with permission, © Springer Nature, 2017).

Computer simulation paradigm based on SRNN

To mimic the rule-dependent computation in the PFC circuit, we set up a computational model for performing the cognitive task. The complete model consisted of three independent components: an input encoder, a SRNN and an output decoder, as shown in Fig. 2A. The role of input encoder was to transform the analog input signal into spikes. We used five leaky LIF neurons for representing the 2AFC task: one neuron for generating cueing or rule inputs, and the other four neurons for encoding distinct sensory cues (e.g., Vision/Left, Vision/Right, Audition/Left, Audition/Right as in [18], Fig. 2B). The role of output decoder was to integrate and transform the spikes from the

SRNN into an analog signal for final (binary) decision. We used two LIF neurons for the 2AFC task. The final voltage values of readout neurons were interpreted as the probabilities of binary decision.

To mimic the behavioral task in rodent experiments, we used the following computer simulation setup:

- 200-ms fixation period, during which only white noise was supplied.
- 100-ms cueing period, during which the spike trains were generated by the cueing encoder and sent to the SRNN.
- 400-ms cueing delay period, during which only white noise was supplied.
- 100-ms stimulus presentation period, during which two spike trains were represented: one from the visual channel, and the other from the auditory channel.
- Response period, during which the cost function was computed at the end of the presentation period, and the network back-propagated the error and computed the gradient.

For each rule condition, we simulated independent Monte Carlo trials based on different initial conditions. In what follows, we present the mathematical details and implementation of each component. The mathematical symbols and notations are summarized in Table 1.

Encoder and decoder The encoder computed the continuous-valued membrane potential voltage trace $\{V(t)\}$ and discrete-valued spike trains $\{S(t) = \sum \delta(t - t_i)\}$ iteratively as follows

$$\tau_{\mathrm{m}} \frac{dV(t)}{dt} = -(V(t-1) - v_{\mathrm{rest}}) + I(t-1) \cdot R \quad (1)$$

$$V(t) = V(t-1) + \frac{dV(t)}{dt} \cdot dt \quad (2)$$

$$S(t) = \Theta(V(t) - v_{\mathrm{th}}) \quad (3)$$

$$V(t) = (1 - S(t)) \cdot V(t) + S(t) \cdot v_{\mathrm{rest}} \quad (4)$$

where $\tau_{\mathrm{m}} = RC$ denotes the time constant for the voltage membrane potential. First, given an input current $I(t)$, the membrane potential voltage $V(t)$ was numerically

Table 1. A summary of symbols and notations.

Symbol	Description
$I_{\text{noise}}(t)$	Noise for baseline current at time step t
$V_{\text{noise}}(t)$	Noise for voltage at time step t
$I_{\text{base}}(t)$	Baseline input current at time step t
$I(t)$	Current at time step t
$V(t)$	Voltage at time step t
$V_{\text{th}}(t)$	Voltage threshold for firing spike at time step t
$A(t)$	Variable component of the firing rate threshold at time step t
$S(t)$	Spike train output at time step t
$S_{\text{cueing}}(t)$	Cueing input spike for SRNN time step t
$S_{\text{sensory}}(t)$	Sensory input spike for SRNN time step t
$S_{\text{rec}}(t)$	Recurrent spike for SRNN at time step t
$S_{\text{readout}}(t)$	Readout spike from SRNN at time step t
$\boldsymbol{W}_{\text{rec}}$	Recurrent connection weights of SRNN
$\boldsymbol{W}_{\text{out}}$	Connection weights of the readout layer
$\boldsymbol{W}_{\text{cueing}}$	Connection weights of the cueing input layer
$\boldsymbol{W}_{\text{sensory}}$	Connection weights of the sensory input layer
$\Theta(\cdot)$	Heaviside step function
$\sigma(\cdot)$	fast sigmoid function
$\odot(\cdot)$	elementwise multiple
ϕ_{exc}	The ratio of excitatory neurons
ϕ_{sfa}	The ratio of spike frequency adaptation (SFA) neurons
R	Resistance of neurons
v_{rest}	Voltage constant at the resting state
v_{th}	Base voltage threshold constant for firing spike
τ_{m}	Time constant of membrane potential
τ_{syn}	Time constant of synapse
τ_{synE}	Time constant of synapse for excitatory neurons
τ_{synI}	Time constant of synapse for inhibitory neurons
τ_{a}	Time constant of SFA
τ_{ref}	Time constant for the refractory period
η	Learning rate
g	Rescale constant for $\boldsymbol{W}_{\text{rec}}$
β	Scale constant for the surrogate gradient
ξ	Scale constant for SFA
α	Random walk coefficient
λ	The coefficient for firing rate regularization
dt	Time step

Table 2. Parameters used in the encoder and decoder.

Symbol	Description
I_{init}	0.0 mA
V_{init}	0.0 mV
v_{rest}	0.0 mV
v_{th}	1.0 mV
R	1 Ω
dt	2 ms
τ_{m}	5 ms
τ_{syn}	20 ms

computed based on the integration of ODE (Eq. 1 and Eq. 2). Next, in Eq. (3), spikes were produced by calculating the distance between $V(t)$ and a preset threshold constant v_{th}, followed by a hard-limiting operation via the Heaviside step function Θ. Once a spike was fired after reaching v_{th}, $V(t)$ was reset to v_{rest} (Eq. 4). Here, for the simplicity of generating inputs, the current I was assumed to be independent of the neuron, implying that there was only an external input, and the internally decayed current from the previous time step was not considered.

The readout decoder employed simple LIF neurons, whose roles were to convert the output spike trains into an informative readout in the vector form. The mathematical equations of the readout decoder are described as follows:

$$\tau_{\text{m}} \frac{dV(t)}{dt} = -(V(t-1) - v_{\text{rest}}) + I(t-1) \cdot R \tag{5}$$

$$V(t) = V(t-1) + \frac{dV(t)}{dt} \cdot dt \tag{6}$$

$$S(t) = \Theta(V(t) - v_{\text{th}}) \tag{7}$$

$$V(t) = (1 - S(t)) \cdot V(t) + S(t) \cdot v_{\text{rest}} \tag{8}$$

$$\frac{dI(t)}{dt} = -I(t-1)/\tau_{\text{mem}} \tag{9}$$

$$I(t) = I(t-1) + \frac{dI(t)}{dt} \cdot dt + S_{\text{external}}(t) \cdot \boldsymbol{W}_{\text{out}} \tag{10}$$

In Fig. 2, the output spike trains $S_{\text{external}}(t)$ from the SRNN was fed into a decoder, characterized by $\boldsymbol{W}_{\text{out}}$ (Eq. 10). In the decoding phase, temporal information was integrated in time (Eq. 9 & Eq. 10). Finally, the analog signals (i.e., voltage $V(t)$) at the final 100 ms were used to represent the choice probability vectors.

Mathematical description of SRNN The structure of SRNN had a similar form as the rate-based RNNs [7], except that each unit of the SRNN consisted of a biologically-constrained LIF neuron. All neurons were connected through a recurrent weight matrix $\boldsymbol{W}_{\text{rec}}$, and the SRNN received an input from the encoder, and produced an output through the output weight matrix $\boldsymbol{W}_{\text{out}}$.

Together, all neurons computed the voltage $V(t)$ and spike trains $S(t)$, which were represented by two vectors $\vec{V}(t)$ and $\vec{S}(t)$ summarizing all neurons' internal states:

$$\tau_{\text{m}} \frac{d\vec{V}(t)}{dt} = -(v_{\text{rest}} - \vec{V}(t-1)) + \vec{I}(t-1) \cdot R + 2\sqrt{dt} \cdot \vec{V}_{\text{noise}}(t) \tag{11}$$

$$\vec{V}(t) = \vec{V}(t-1) + \frac{d\vec{V}(t)}{dt} \cdot dt \tag{12}$$

$$\vec{A}(t) = \varphi \cdot \vec{A}(t-1) + (1-\varphi) \cdot \vec{S}(t) \tag{13}$$

$$\vec{V}_{\text{th}}(t) = v_{\text{th}} + \xi \cdot \vec{A}(t) \odot \vec{A}_{\text{mask}} \tag{14}$$

$$\vec{S}(t) = \Theta(\vec{V}(t) - \vec{V}_{\text{th}}(t)) \tag{15}$$

$$\vec{V}(t) = (1 - \vec{S}(t)) \cdot \vec{V}(t) + \vec{V}(t) \cdot v_{\text{rest}} \tag{16}$$

$$\vec{S}(t), \vec{V}(t) = \text{Refractory_Clamp}\{\vec{S}(t), \vec{V}(t), \tau_{\text{ref}}\} \tag{17}$$

$$\tau_{\text{syn}} \frac{d\vec{I}(t)}{dt} = -\vec{I}(t-1) \tag{18}$$

$$\vec{I}_{\text{base}}(t) = \alpha \cdot \vec{I}_{\text{base}}(t-1) + \vec{I}_{noise}(t) \tag{19}$$

$$\begin{aligned} \vec{I}(t) = {} & \vec{I}(t-1) + \frac{d\vec{I}(t)}{dt} \cdot dt \\ & + \vec{S}_{\text{cueing}}(t) \cdot \boldsymbol{W}_{\text{cueing}} + \vec{S}_{\text{sensory}}(t) \cdot \boldsymbol{W}_{\text{sensory}} + \vec{S}_{\text{rec}}(t) \cdot \boldsymbol{W}_{\text{rec}} \\ & + \vec{I}_{\text{base}}(t) \end{aligned} \tag{20}$$

The spike generation (Eqs. 11-20) followed the basic mechanism of LIF. In addition, we imposed a hard refractory period right after a neuron fires a spike (Eq. 17), and SFA (Eqs. 13-14).

The current $I(t)$ consisted of multiple contributing components: (i) the spike trains derived from cueing and sensory inputs in the encoder were converted into the current via respective connection weights $\boldsymbol{W}_{\text{cueing}}$ and $\boldsymbol{W}_{\text{sensory}}$; (ii) the current contributed from recurrent spikes $\vec{S}_{\text{rec}}$ scaled by the recurrent connections $\boldsymbol{W}_{\text{rec}}$; (iii) baseline current $\vec{I}_{\text{base}}$.

The noise provided a more realistic environment in the noisy nervous system, making

the trained SRNN more robust. Two sources of noise were considered in this model, which ensured that even in the absence of external inputs, the SRNN still fired and maintained a reasonable level of neuronal firing rate. The first source of noise was the voltage noise: $\vec{V}_{noise}$ sampled from a normal distribution. The second source of noise was the baseline current noise: $\vec{I}_{noise}$, which was implemented through a random walk (Eq. 19). At every time step, $\vec{I}_{noise}$ was sampled from a truncated normal distribution.

Our SRNN has several important features: (i) the SRNN consists of excitatory and inhibitory neurons that follow Dale's principle; (ii) the SRNN implements an efficient SuperSpike algorithm [16] to alleviate the vanishing gradient problem frequently encountered in the RNN; (iii) the SRNN employs a SFA strategy to allow error back propagation through time (BPTT) during the delay period [25]; and (iv) the SRNN uses a regularized cost function that penalizes the number of spikes (L_0 regularization) in the neuronal population (thereby imposing a sparsity constraint).

Network initialization We initialized the neuronal states and neuronal baseline firing rates using the following procedure. First, we randomized the initial currents and voltages for every neurons within a specific yet reasonable range at each trial. The range was chosen to optimize the robustness of the model. Second, we assigned each neuron with different baseline firing rates. We injected the constant current $\vec{I}_{\text{base}}$ to neurons at every time step, which produced the resting firing rate. We chose the distribution of $\vec{I}_{\text{base}}$ from a truncated normal distribution (Table 3).

Imposing biological constraints

Similar to the previous work [7], we imposed Dale's principle onto the SRNN. Specifically, cortical neurons have either purely excitatory or inhibitory effects on postsynaptic neurons, and the excitatory cortical neurons outnumber inhibitory cortical neurons by a ratio of around 4:1. The synaptic connection w_{jk} from neuron j to neuron k is excitatory (or inhibitory) if $w_{jk} > 0$ (or $w_{jk} < 0$). Furthermore, the algebraic sign of w_{jk} does not change during learning or neural plasticity.

According to the excitatory (E) and inhibitory (I) populations, the recurrent weight matrix $\boldsymbol{W}_{\text{rec}}$ was decomposed into four blocks: $\boldsymbol{W}_{\text{EE}}, \boldsymbol{W}_{\text{EI}}, \boldsymbol{W}_{\text{IE}}, \boldsymbol{W}_{\text{II}}$. The elements in

$\boldsymbol{W}_{\mathrm{EE}}$ and $\boldsymbol{W}_{\mathrm{EI}}$ were all positive, whereas the elements in $\boldsymbol{W}_{\mathrm{II}}$ and $\boldsymbol{W}_{\mathrm{IE}}$ were all negative. Given a randomly initialized $\boldsymbol{W}_{\mathrm{init_rec}}$, the absolute values of the entries were extracted for resetting the positive and negative property. We assumed no self-connection, namely, we used $\boldsymbol{W}_{\mathrm{rec_mask}}$ to set the diagonal values as zeros. Finally, the reset matrix was multiplied with a diagonal matrix $\boldsymbol{W}_{\mathrm{Dale}}$ to set the ratio of excitatory-inhibitory neurons to 4:1 as follows:

$$\boldsymbol{W}_{\mathrm{rec}} = \left|\boldsymbol{W}_{\mathrm{init_rec}}\right| \odot \boldsymbol{W}_{\mathrm{rec_mask}} \cdot \boldsymbol{W}_{\mathrm{Dale}} \tag{21}$$

$$\boldsymbol{W}_{\mathrm{rec}} = \left|\begin{bmatrix} w_{1,1} & w_{1,2} & \dots & w_{1,n-1} & w_{1,n} \\ w_{2,1} & w_{2,2} & \dots & w_{2,n-2} & w_{2,n} \\ \vdots & & \ddots & \vdots & \vdots \\ w_{n-1,1} & w_{n-1,2} & \dots & w_{n-1,n-1} & w_{n,n} \\ w_{n,1} & w_{n,2} & \dots & w_{n,n-1} & w_{n,n} \end{bmatrix}\right| \tag{22}$$

$$\odot \begin{bmatrix} 0 & 1 & \dots & 1 & 1 \\ 1 & 0 & \dots & 1 & 1 \\ \vdots & & \ddots & & \vdots \\ 1 & 1 & \dots & 0 & 1 \\ 1 & 1 & \dots & 1 & 0 \end{bmatrix} \cdot \underbrace{\begin{bmatrix} d_e & & & & \\ & d_e & & & \\ & & \ddots & & \\ & & & d_i & \\ & & & & d_i \end{bmatrix}}_{\mathrm{Exc:Inh}=\phi_{\mathrm{exc}}:(1-\phi_{\mathrm{exc}})} \tag{23}$$

where $\odot$ denotes the element-wise multiplication of two matrices; d_e and d_i were positive and negative, respectively.

To keep the balance of excitation and inhibition [7,26], the total connection strengths in $\boldsymbol{W}_{\mathrm{rec}}$ for the E and I populations should be roughly equal. Specifically, we optimized $\boldsymbol{W}_{\mathrm{Dale}}$ such that the sum of all d_e entries was identical to the sum of all d_i entries, and the resultant $\boldsymbol{W}_{\mathrm{rec}}$ was approximately E-I balanced.

All the connection weights were first initialized via the Glorot initialization algorithm [27]. Next, we rescaled the initial $\boldsymbol{W}_{\mathrm{init_rec}}$ based on its eigenvalue spectra [1,28]. Let $\rho = \max\{|\lambda_1|, \dots, |\lambda_n|\}$ denote the largest absolute value of eigenvalues of $\boldsymbol{W}_{\mathrm{rand_rec}}$ (a randomly generated matrix); we scaled $\boldsymbol{W}_{\mathrm{rand_rec}}$ by g/ρ

such that as

$$\boldsymbol{W}_{\text{init_rec}} = g/\rho \cdot \boldsymbol{W}_{\text{rand_rec}} \tag{24}$$

where $g \in [1.2, 1.5]$.

Synaptic time constant The synaptic time constant τ_{syn} in our SRNN was separated into τ_{synE} for excitatory neurons and τ_{synI} for inhibitory neurons. In general, as fast-spiking interneurons, inhibitory neurons should have higher firing rates than excitatory neurons at the population level. In order to modulate this difference, τ_{synI} was assigned by a greater time constant, since a slower decaying of synaptic current implies a larger possibility for generating spikes.

Spike frequency adaptation (SFA) SFA is a dynamic mechanism that some neurons's spiking threshold is adaptive, increasing when spiking. Specifically, based on the baseline spiking threshold v_{th}, SFA neuron k had an extra item A_k that adapted in time, under the constant scale of ξ, corresponding to the spike frequency. First, A_k decayed when time passing with the time constant τ_{a}, as illustrated by the ODE in Eq. (25). Second, once the neuron k fired, A_k increased one-shot instantaneously. Equation (26) gives the discrete integration formula for explaining the phenomena at specific time step; φ in Eq. (27), describes the decaying of SFA at the level of integration.

$$\tau_{\text{a}} \frac{dA_k(t)}{dt} = -A_k(t-1) + v_{\text{th}} \tag{25}$$

$$A_k(t) = \varphi A_k(t-1) + (1-\varphi) S_k(t-1) \tag{26}$$

$$\varphi = \exp\left(-\frac{dt}{\tau_a}\right) \tag{27}$$

$$V_k^{\text{th}}(t) = \xi A_k(t) + v_{\text{th}} \tag{28}$$

We used A_{mask} to separate SFA and non-SFA neurons and used ϕ_{sfa} to denote the ratio of SFA neurons.

Recently, it has been shown that SRNNs with SFA can significantly increase the longer short-term memory capability [24, 25], showing comparable capabilities as the RNN with long short-term memory (LSTM) units. This is because SFA-enabled neurons, when spiking continuously, obtain a higher instantaneous voltage potential but

hardly generate spikes. Therefore, in comparison with non-SFA neurons, the time for their membrane potential to decay to the resting state becomes longer, implying that the memory is extended by SFA.

SRNN training

The output of the SRNN was a two-dimensional (2D) continuous-valued vector extracted from the decoder at the final time step during the stimulus presentation period. Next, we normalized the 2D vector into a predicted probability distribution vector $\hat{\mathbf{y}} = \{\hat{y}_i\}$, with $\mathbf{y}$ denoting the corresponding target vector output. We used the mean-squared error (MSE) as the cost function that summed over all N single trials

$$\begin{aligned} \mathcal{L} &= \sum_{\ell=1}^{N} \mathcal{L}_\ell \\ &= \sum_{\ell=1}^{N} \sum_{t=1}^{T_{\text{loss}}} (y_{l,t} - \hat{y}_{l,t})^2 \end{aligned} \tag{29}$$

where $T_{\text{loss}} = 5$ denotes the number of time steps used for cost calculation during sensory stimulus presentation period.

We incorporated firing rate regularization into the cost function, which penalized the mean firing rate for global neuronal population. This regularization avoided the excessive firing frequency, also prevented the firing rate of excitatory neurons higher than that of inhibitory neurons. Specifically, the regularized cost function is written as

$$\mathcal{L}_{\text{reg}} = \mathcal{L} + \lambda R^{\text{fr}} \tag{30}$$

$$R^{\text{fr}} = \frac{1}{N T_{\text{binned}}} \sum_{l=1}^{N} \sum_{t=1}^{T_{\text{binned}}} (\mathbf{r}_{l,t})^2 \tag{31}$$

where $\lambda > 0$ is the regularization coefficient, and T_{binned} is the time step length after binning spikes with bin size of 5.

Let w_{jk} denote the connection strength from neuron j to k; the parameters of the SRNN were updated iteratively according to stochastic gradient descent

$$w_{jk}^{(\ell)} = w_{jk}^{(\ell-1)} - \eta \Big(\frac{\partial \mathcal{L}_{\text{reg}}}{\partial w_{jk}} \Big)^{(\ell-1)} \tag{32}$$

where $\eta > 0$ is a learning rate parameter, and $\frac{\partial \mathcal{L}_{\text{reg}}^{(\ell-1)}}{\partial w_{jk}}$ denotes the gradient evaluated on the parameter w_{jk} from iteration $\ell - 1$.

We used the BPTT algorithm to train the parameters of SRNN. However, the back-propagating gradient flow shrinks quickly in the SRNN when it passes the Heaviside step function Θ (Eq. 11) while adapting the gradient [17]. This is because the gradient is equal to zero in the absence of spikes. Therefore, according to the chain rule, the gradient propagated from its previous layer would suffer from a great loss. To reduce the loss during back-propagation, we replaced the gradient in the chain rule by a surrogate gradient known as SuperSpike surrogate gradient [16].

SuperSpike surrogate gradient The SuperSpike is a nonlinear voltage-based three-factor learning rule, which is capable of training multilayer networks of deterministic LIF neurons to perform nonlinear computations on spatiotemporal spike patterns [16]. Specifically, the SuperSpike surrogate gradient was computed as follows

$$\frac{\partial S_k(t)}{\partial w_{jk}} \approx \sigma'\left(V_k(t)\right) \frac{\partial V_k(t)}{\partial w_{jk}} \tag{33}$$

$$\sigma'\left(V_k(t)\right) = \left(1 + \left|\beta\left(V_k(t) - V_{\text{th}}(t)\right)\right|\right)^{-2} \tag{34}$$

where β is a scaling parameter that defines the sharpness of the approximate derivative: the greater the β, the sharper is the derivative. In Eq. (33), the Heaviside step function $\Theta(V(t))$ was approximated by a continuous sigmoid-shaped function $\sigma(V(t))$ when calculating the surrogate gradient. Therefore, at every time step we obtained a smoothed gradient of the Heaviside step function, and prevented BPTT from gradient vanishment.

The computation of SuperSpike gradient is interpreted as Hebbian coincidence detection [16]. Specifically, the gradient computation of $\frac{\partial S_k(t)}{\partial w_{jk}}$ consists of the product of two derivative terms (Eq. 33), the first-order derivative term $\sigma'\left(V_k(t)\right)$ corresponds to the dynamics of k-th neuron, which can be regarded as the postsynaptic activity. The second partial derivative term $\frac{\partial V_k(t)}{\partial w_{jk}}$ approximates the concentration of neurotransmitters at the synapse w_{jk}, which can represent the presynaptic activity. Thus, training based on SuperSpike gradient can be regard as an analogue of Hebbian learning.

Back-propagation-through-time in SRNN To implement BPTT, similar to the prior work [17], we approximated the SRNN dynamics to update the voltage and current as follows:

$$
\begin{aligned}
V_k(t) &= \exp\left(\frac{-dt}{\tau_{\text{mem}}}\right) \cdot \Big[(V_k(t-1) - v_{\text{rest}}) + I_k(t-1) \cdot R\Big] - S_k(t-1) \cdot (V_{\text{th}}(t) - v_{\text{rest}}) \\
I_k(t) &= \exp\left(\frac{-dt}{\tau_{\text{syn}}}\right) \cdot I_k(t-1) \\
&\quad + \sum_j S_j^{\text{cueing}}(t-1) \cdot w_{jk}^{\text{cueing}} + \sum_j S_j^{\text{sensory}}(t-1) \cdot w_{jk}^{\text{sensory}} \\
&\quad + \sum_j S_j(t-1) \cdot w_{jk}^{\text{rec}} + I_k^{\text{base}}(t-1)
\end{aligned}
$$

When computing the gradients of S_k, V_k and I_k in the k-th neuron with respect to the recurrent weight w_{jk}^{rec}, we obtained the following equations:

$$
\frac{\partial S_k(t)}{\partial w_{jk}^{\text{rec}}} = \left(1 + |\beta\,(V_k(t) - v_{\text{rest}})|\right)^{-2} \frac{\partial V_k(t)}{\partial w_{jk}^{\text{rec}}} \tag{35}
$$

$$
\begin{aligned}
\frac{\partial V_k(t)}{\partial w_{jk}^{\text{rec}}} &= \exp\left(\frac{-dt}{\tau_{\text{mem}}}\right) \cdot \left(\frac{\partial V_k(t-1)}{\partial w_{jk}^{\text{rec}}} + \frac{\partial I_k(t-1)R}{\partial w_{jk}^{\text{rec}}}\right) \\
&\quad - (V_{\text{th}}(t) - v_{\text{rest}}) \cdot \frac{\partial S_k(t-1)}{\partial w_{jk}^{\text{rec}}}
\end{aligned} \tag{36}
$$

$$
\frac{\partial I_k(t)}{\partial w_{jk}^{\text{rec}}} = \exp\left(\frac{-dt}{\tau_{\text{syn}}}\right) \cdot \frac{\partial I_k(t-1)}{\partial W_{jk}^{\text{rec}}} + S_j(t-1) \tag{37}
$$

We employed the Adam algorithm [29] to update the recurrent weights $\boldsymbol{W}_{\text{rec}}$ and output weights $\boldsymbol{W}_{\text{out}}$; additionally, we used an early-stopping criterion to alleviate overfitting.

Identification of rule-specific neuronal tuning

Given single-trial spike train simulations of each neuron, we displayed the spike rasters and computed the peri-stimulus time histogram (PSTH) using a 8-ms bin size (i.e., 4 bins). To obtain the smooth tuning profile, the PSTH was convolved with a Gaussian kernel to create a spike density function, which was further Z-scored by subtracting the mean firing rate during the delay period (for both rules) and dividing the standard deviation (SD) over the same period. The neurons with Z-scored peak firing rate above a threshold were considered to have sharp tuning. We first screened the neurons with one or more sharp candidate peaks during the delay period for the subsequent analysis.

Table 3. Hyperparameters used in the SRNN.

Symbol	Description
$I_{\text{base}}(t=0)$	$0.2 \times \mathcal{N}(\mu = 2, \sigma^2 = 0.01)$, truncated in [0.3, 0.5] (unit: nA)
$I_{\text{noise}}(t)$	$0.2 \times \mathcal{N}(\mu = 0, \sigma^2 = 4)$ (unit: nA)
v_{rest}	-65 mV
v_{th}	-50 mV
ϕ_{exc}	80%
ϕ_{sfa}	25%
dt	2 ms
β	1000 mV
R	10^7 Ω
τ_{mem}	20 ms
τ_{synE}	32 ms
τ_{synI}	65 ms
τ_{a}	400 ms
τ_{ref}	6 ms
η	3×10^{-4}
g	1.5
ξ	1.6
α	1
λ	10^{-7}

Neurons with very low mean firing rates (< 2 Hz) during the delay period were not considered in the tuning analysis. We used a similar criterion as [18] to identify the genuine tuning peaks. In addition to the minimum firing rate criterion, these candidate peaks needed to have Z-score values of greater than a threshold (e.g., 2.33 for $p < 0.01$). We used at least $p < 0.01$ as the significance criterion. We further categorized the neurons with putative tuning peaks depending on the number of peaks. The majority of identified neurons with rule-specific tuning had 1-2 peaks.

Characterization of sequential activation

We computed the sequentiality index (SI) to characterize the sequential activation of the population response of excitatory neurons during the delay period. The SI is defined as the sum of the entropy of the peak response time distribution of the recurrent neurons and the mean log ridge-to-background ratio of the neurons, where the ridge-to-background ratio for a given neuron is defined as the mean activity of the neuron inside a small window around its peak response time divided by its mean activity outside this window [21, 30]. The computer code for computing the SI was available online (https://github.com/eminorhan/recurrent-memory/blob/087a4a67b684ca2e8b6d8102c4221c88105cc16f/utils.py#L16). To test for

the significance of median SI differences between two conditions, we used a rank-sum test throughout the study.

Dimensionality reduction

To investigate the neuronal population response, we used the principal component analysis (PCA) to extract the principal components of population responses during the delay period. To compute the neural representation or trajectories in the state space, we projected the data onto the subspaces of dominant principal components (PCs). and obtained the low-dimensional latent state trajectory $\mathbf{x}(t)$. Furthermore, we defined the network kinetic energy to characterize the dynamics of SRNN [31]

$$\mathcal{K}(\mathbf{x}) = \frac{1}{2}\|\dot{\mathbf{x}}(t)\|^2 \tag{38}$$

A low value of $\mathcal{K}(\mathbf{x})$ indicates the state reaches a fixed-point region with low or zero momentum.

Population decoding

We used a support vector machine (SVM) to read out the neuronal population responses during delay period. The goal of the binary SVM classifier is to construct the classification boundary by a separating hyperplane with maximum margin. Here we used a linear kernel for binary classification, and trained the linear SVM with the sequential minimal optimization algorithm. The performance of linear SVM was relatively insensitive to the choice of hyperparameter (as opposed to the nonlinear SVM based on a radial basis function kernel).

Upon completion of SRNN training, we produced a total of 100 trials (evenly for each rule/stimulus configuration) for classification analysis. We temporally binned the spike trains (10 ms bin size) during the delay period. To train the SVM classifier, we used the spike counts of individual neurons or PCs as the features. The purpose of PCA was to reduce the number of features and to avoid overfitting. We randomly split 100 trials into two groups, 90% used for training, and 10% used for testing. We conducted 10-fold cross-validation and then computed the average test accuracy. The default classification threshold was 0.5.

Software implementation

The RNN implementation was built upon the Python package `Norse` (https://github.com/norse/norse), based on PyTorch and deep learning-compatible spiking neural network components, including network architectures and optimization algorithms. Our custom software implementation can be found in https://github.com/Jakexxh/SNN_PFC-MD.

Results

Training SRNN for the 2AFC task

In the default setup ($n = 500$), we used 400 excitatory neurons and 100 inhibitory neurons in the SRNN. We trained the SRNN on the Oracle's cloud-based GPU-accelerated virtual machine (NVIDIA GPU P100, 16 GB GPU memory, 12 CPU cores, 78 GB CPU memory) within 10,000 epochs, the performance gradually reached 100% (Fig. S1A,B). Based on different random initial conditions, we repeated the training procedure 10 times and obtained 10 realizations of trained SRNN. Each trained network was treated as one set of independent data. In total, we examined the task representations of 5000 simulated (4000 excitatory and 1000 inhibitory) neurons, and used them in the subsequent analyses. The initial distribution of synaptic weights (absolute value) was close to a uniform distribution. After learning, the weight distribution had a long-tailed behavior, and many of synaptic strengths were close to zeros (Fig. S1C).

Rule-specific neuronal representations

Upon completion of learning, we first examined the neuronal responses at the single-unit level. We identified task-modulated excitatory neurons that showed sharp peaks during the delay period. We found some excitatory neurons showed rule-specific tunings, with typical differential peaks or peak locations between two rules (Fig. 3A,B). As seen in the spike raster examples, some neurons showed sharp tuning at specific timing for coding one rule, but not the other. The number of peaks was comparable between two rules. A smaller percentage of units were tuned to both rules, with distinct temporal offset in tuning peak (Fig. 3C). This emergent property was reminiscent of the experimental finding in rodent PFC neuronal recordings (Fig. 1B). The ratio of task-modulated or rule-specific tuning excitatory neurons was around 12.5-15% (mean: 13.1% averaged over 10 realizations), qualitatively similar to experimental findings [18], and the ratio of excitatory neurons representing rule 1 and rule 2 was roughly equal (6.55% vs. 6.53%). In addition, the averaged firing rates of excitatory and inhibitory neurons during the delay period were 4.80 ± 0.36 Hz and 7.76 ± 2.42 Hz, respectively.

Interestingly, we observed a slightly increase in the firing rate of excitatory neurons during the delay period as compared to the cue period, yet a dramatic firing rate decrease in inhibitory neurons (Fig. 3D).

Furthermore, we sorted the rule-specific excitatory neurons according to the peak timing at each rule (Fig. 3E,F). Together, their trial-averaged population responses (visualized as a normalized heat map) showed an emergence of neural sequence (comparing with Fig. 1C). Notably, neurons contributed to different roles in sequence representation at different rules. To quantified the variability of the sequential activation, we further computed the SI measure (Methods) for rule-specific excitatory neurons as well as for the complete excitatory populations (Fig. S2). We found that the rule-tuned neurons had a greater SI median statistic than the whole excitatory neuronal population (mean±SD: 4.50 ± 0.06 vs. 4.36 ± 0.08; $p < 10^{-5}$, rank-sum test). We also compared the SI statistics between trained and untrained network based on the excitatory neuronal population (Fig. 3G). To obtain the chance level, we fed the white noise input to the trained SRNN and recomputed the SI based on excitatory neuronal activation during the delay the period. The chance-level SI was significantly less than the task condition ($p = 0.00015$, rank-sum test).

In addition, we observed rhythmic firing in spiking activity of a subset of inhibitory neurons during the delay period. This was also independent of their task-tuned properties. A closer look at their PSTHs or auto-correlograms revealed strong beta (15-30 Hz) and gamma (30-40 Hz) oscillations at the spiking activity (Fig. 4). Some inhibitory neurons also showed differential rule-specific firing, in terms of oscillatory frequency or phase. However, we did not find any correlation between the power of these rhythmic activity and task performance. In total, we found 11.6% (116/1000) inhibitory neurons with strong beta or gamma oscillations.

Neuronal coupling and connectivity

Next, we examined the pairwise neuronal coupling, especially those pairs with large (in absolute value) recurrent connection weights of the trained SRNN. In computing the spike-time cross-correlation between E-E, E-I, I-E and I-I neuronal pairs, we focused on the identified rule-tuned neurons as either the trigger or the target. Interestingly, we

found many E-E pairs of rule-tuned neurons representing the same rule showed a high peak with a small time lag (2-8 ms) in their cross-correlograms (Fig. 5A). For E-I or I-E pairs, we often found a relatively large trough in their cross-correlograms (Fig. 5B,C). For I-I pairs, the cross-correlogram could have different profiles, depending on whether the rhythmically modulated units were the target (Fig. 5D, left panel), or trigger (Fig. 5D, middle panel), or both (Fig. 5D, right panel).

We further investigated whether single neuronal activities were related to any cluster within the trained network. For each neuron, we computed the trial variances at three different stages (cue period, delay period, and stimulus presentation period, each period with two conditions) and further embedded the 6-dimensional vector in a 2D space using t-distributed stochastic neighbor embedding (tSNE) algorithm [32]. Interestingly, we found functionally distinct clusters in the embedded space between excitatory and inhibitory populations, as well as within excitatory subpopulations (Fig. S3).

Neural trajectory analysis

At the population level, the high-dimensional SRNN dynamics (Fig. 6A) during the delay period can be visualized via dimensionality reduction. Specifically, we conducted PCA and plotted the low-dimensional neural trajectory in the PC space (Fig. 6B). At the PC subspace, we observed the kinetic energy reached the maximum around 100 ms, and then decayed to around zero after 200 ms (Fig. 6C).

Next, we ran a population decoding analysis based on 400 excitatory neuronal activities during the delay period. To test the rate code hypothesis, we used an 20-ms non-overlapping moving window to compute the trial-by-trial spike counts of individual neurons. We further projected them onto the PCA subspace, and then decoded the rules from the first two PCs. We found out that 10-fold cross-validated (on 10% held-out trials) decoding accuracy reached 70-75% at the first 100 ms, and then gradually increased as the information accumulated during the delay period; finally, the decoding accuracy saturated around 300 ms (Fig. 6D). The performance plateau can be explained by the flat temporal profile of kinetic energy in the last 150-200 ms of delay period. These results also suggest that the rule information was more likely to be encoded by individual neurons' spike timing at a finer timescale in addition to their firing rates.

Error trials induced by rule encoding uncertainties

Thus far we have limited our analysis on correct trials performed by the trained SRNN. In animal experiments, behavioral errors may arise from executive error, sensory error, or both. The executive error is related to the rule encoding, whereas sensory error is related to the target cue perception. To emulate executive-type error trials, we introduced a mixed representation of two continuous rule inputs by varying their proportional ratios q and $1-q$ (q proportion of rule signal 1 plus $(1-q)$ proportion of rule signal 2), and then converted the continuous input via the encoder into a discrete spike train. We fed the spike train that represented the mixed rule signal to the trained SRNN. As expected (Fig. 7A), the network produced an error trial with the highest probability when the ambiguity level was greatest (i.e., $q = 0.5$), and the average network performance gradually decreased when the degree of ambiguity increased.

We further examined the neural representations of single units during error trials. Generally, we found that the rule-specific tuned units during correct trials changed their firing profiles: (i) units tuned to rule 1 changed preferred tuning to rule 2, or vice versa (e.g., Fig. 7B); (ii) units tuned to both rules changed their peak-firing timing (e.g., Fig. 7C). At the population level, we also projected the neuronal responses of error trials onto the PC subspace; the error-trial trajectory showed a clear deviation from the correct-trial pattern. For instance, the trial mistaken classified as rule 2 started a trajectory closer to rule 1, and then wandered in the neural state space before settling in a region closer to rule 2 (Fig. 7D).

Impact of elongated delay period

Motivated from the experimental manipulation [18], we extended the 400-ms delay period to 800 ms in the testing phase, and ran the SRNN simulations for the complete 800-ms period. We further examined the neuronal responses in the elongated 400-ms period. Interestingly, we found that some non-task-modulated neurons developed a late peak in the 800-ms delay condition (Fig. 8B); in contrast, the task-modulated neurons that showed an early peak in the first 400-ms delay period preserved the peak timing even in the 800-ms delay (Fig. 8A). The latter case of neuronal responses implies the time-invariant temporal coding. Additionally, we observed a reduction in task

performance with an increasing delay period (Fig. 8C), indicating the encoded rule information were gradually lost with increasing time.

Impact of dropped out cortical connections, network connectivity and E/I balance

Furthermore, we investigated the fault tolerance of SRNN with respect to the cortical connectivity. We randomly dropped out a small percentage of synaptic weights in $\boldsymbol{W}_{\mathrm{rec}}$ and set them zeros. We ran the network simulations using the modified synaptic weight matrix and tested its impact on neural representations and task performance. We observed the sparsity affected the rule-specific tuning in excitatory neurons. Some observe peak timing disappeared with an increasing sparsity in $\boldsymbol{W}_{\mathrm{rec}}$ (Fig. 9A,B). With an increased sparsity level of recurrent weight matrix, we also observed a decrease in task performance (Fig. 9C) as well as cross-validated SVM decoding accuracy (Fig. 9D). Notably, for the same level of sparsity, the task performance was more robust than the SVM decoding strategy based on the population spike count readout during the delay period.

To examine this effect of cell-type-specific neuronal connectivity on the neural representation and task performance, we scaled up or down the $\boldsymbol{W}_{\mathrm{EE}}$ connectivity (by multiplying a scaling factor), and measured the changes in firing rates of E and I neuronal populations (Fig. 10A). As expected, the scaling factor affected the E/I firing rate and E/I balance. We further measured the impact on task performance in the trained SRNN (Fig. 10B). When the E/I was imbalanced, the task performance became poor. In the extreme condition where the overall excitation dominated over inhibition, the performance reduced to a chance level.

As a comparison, we also scaled up or down the $\boldsymbol{W}_{\mathrm{IE}}$ connectivity and reported the firing rate changes. As the scaling factor increased, the global excitation and inhibition both went down, but their relative levels were similar (Fig. 10C). In contrast to the EE connections, the IE connections showed a higher degree of tolerance to the scaling factor (Fig. 10D), indicating the robustness to inhibitory weight perturbation.

At the single-unit level, we found that the rule-tuned units decreased their peak firing rates according to the network E/I balance (Fig. S4). Specifically, we observed

that the modified EE or IE connections generated either conflicting or ambiguous rule tuning (Fig. S4A), or diminished rule tuning (Fig. S4B). Notably, our results were similar the reports shown in optogenetic PFC experiments [18].

Results in the 4AFC task

Finally, we adapted our computer experiments from the 2AFC task to the 4AFC task. Accordingly, we used four neuronal outputs to represent the four possible choices, and the decision was selected based on a softmax function. Since the computer task difficulty was increased, we thereby increased the size of SRNN from $n = 500$ to $n = 800$. We noticed that the the final task performance of the trained SRNN slightly degraded ($\sim$95-98%); however, the rule-tuned neural representations still preserved, at both single unit and population levels (Fig. S5).

Discussion

The context or rule-dependent task behaviors have been modeled in the literature [2, 7]. Mante *et al.* trained a rate-based RNN model of selection and integration to simulate the the rule-dependent computation in the monkey PFC [2]. The rate-based RNN model qualitatively reproduces the monkey's behavior as well as the PFC population responses. There are several major distinctions between our model and theirs. First, our model is a SRNN, which uses a different training algorithm. Second, our model imposes biological constraints onto the recurrent weight connections (i.e., Dale's principle). Song *et al.* extended the rate-based RNN model by imposing similar biological constraints, but their learning framework was fundamentally different from ours [7]. Notably, the distinction of excitatory and inhibition neurons in the SRNN enabled us to examine the impact of cell type-specific connectivity and E/I balance. Third, we introduced an additional biological constraint known as SFA. This also turned out to be important for training SRNN efficiently. Fourth, we imposed an L_0 regularization constraint on the sparsity of excitatory neuronal firing (i.e., the total sum of the spikes), as well as a soft L_1 regularization constraint on the total sum of the absolute connection weights [7]. Notably, we have also independently implemented a rate-based excitatory-inhibitory RNN model as in [7] (results not shown). However, we could not reproduce the

rule-specific tuning at the single-cell level, demonstrating the advantages of the spiking network model.

Our modeling work was strongly motivated based on the previously published data [18]. Our central result is that the trained SRNN performing a 2AFC task can exhibit neural representations and dynamics that resemble the experimental findings of mouse PFC recordings. The resemblance between the SRNN and experimental data was manifested at the levels of single neurons and population responses. Although we didn't model the neural sequence explicitly, the neural sequence representation emerged from the trained SRNN for the 2AFC task. Recent work [30] has suggested that both sequential and persistent activity are part of a spectrum that emerges naturally in the trained rate-based RNN under different conditions; and many factors, such as intrinsic circuit properties, temporal complexity of the task, Hebbian synaptic plasticity, delay duration variability, and structured dynamic inputs, can affect the neural representation of recurrent neurons in the trained network. Our SRNN modeling also confirmed similar findings. We found that the change in sparsity or strength of recurrent weight connections can affect neural representations under various test conditions.

Additionally, our trained SRNN produced emergent beta and gamma oscillatory activity, which was in line with the monkey PFC recordings during working memory tasks [33]. According to the dynamic coding hypothesis [34,35], the oscillatory activity is created by local feedback inhibition shared by local clusters of pyramidal neurons. During attractor activations, the recurrent connections and specific synaptic potentiation produce a slight excitatory bias in the subset of neuronal assemblies. Once they spike, they further activate a new wave of feedback inhibition and turn down the rest of neurons. Overall, our modeling study provides a computational framework to understand a number of PFC findings in cognitive control and working memory. For instance, we observed diverse (dynamic, persistent, and oscillatory) responses of single-unit activity during the delay period. The emerged oscillatory dynamics at the spiking activity seen in the SRNN suggest its computer modeling advantages at a fine temporal resolution, which are missing in the rate-based models.

In general, RNN modeling can provide a computational platform to investigate neural representations of cortical circuits in a wide range of cognitive tasks at a fine timescale. For instance, Yang *et al.* trained a rate-based RNN model for performing 20

cognitive tasks that depend on working memory, decision-making tasks, categorization, and inhibitory control [8]. They found that after training, functionally distinct clusters emerged among recurrent neurons that were functionally specialized for different cognitive processes. Learning often gave rise to compositionality of task representation. In addition, the trained network developed mixed task selectivity similar to recorded PFC neurons after learning multiple tasks sequentially. Extension of our SRNN in a continual learning setting [8, 36, 37], will be the subject of future investigation.

The PFC plays a key role in cognitive flexibility, and its neural representation and working memory function are highly dependent on its interaction with the mediodorsal thalamus (MD) [5, 18, 19, 38]. Recently, the MD has been shown to play a modulatory role in cognitive control by augmenting effective connectivity between PFC neurons. In parallel to their experimental circuit dissection, computational models have been developed to account for their interactions [18, 19]; extension of our SRNN framework to modeling biologically-informed PFC-MD network will be the subject of future investigation.

Finally, while our proposed SRNN is biologically plausible, its training procedure relies on supervised learning and back-propagation. However, it remains arguable whether the brain uses back-propagation to perform synaptic modification [39]. In contrast to the error-correcting learning mechanism, Hebbian plasticity and more specifically spike-timing-dependent plasticity (STDP), has become a well-established mechanism for learning. The STDP provides a biologically plausible unsupervised learning mechanism that locally modifies synaptic weights based on the degree of temporal correlations between the pre- and post-synaptic spike events [14, 40–45]. Therefore, applying temporally asymmetric Hebbian learning for SRNN to model neural sequences represents another important future research direction [46, 47]. In our model, the hyperparameters of intrinsic properties (such as the membrane time constant, firing threshold, and reset potential) were fixed. Recently, it has been shown that joint optimization of synaptic weights (i.e., connectivity patterns) and membrane-related parameters (i.e., intrinsic neuronal properties) of SRNN may help perform complex tasks that require information integration and working memory. Future work will further improve the SRNN and validate it in a realistic and complex task setting.

Acknowledgments

We thank current and former members of the Chen lab for helpful discussion.

Funding: This work was partially supported by the NSF-CBET grant 1835000 (ZSC), NIH R01-NS100065 (ZSC), R01-MH118928 (ZSC). Cloud computing resources are partially supported by the Research Award provided by Oracle for Research. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Data Availability: All custom computer software files are available at https://github.com/Jakexxh/SNN_PFC-MD and https://www.cn3lab.org/software.html.

Author Contributions

Conceived and designed the experiments: ZSC, XX. Supervised the project: ZSC. Provided and collected experimental data: MMH; Performed the computer experiments: XX. Analyzed the data: XX. Contributed the software: XX. Wrote the paper: ZSC, XX.

References

1. Sussillo D, Abbott LF. Generating coherent patterns of activity from chaotic neural networks. Neuron. 2009;63(4):544–557.

2. Mante V, Sussillo D, Shenoy K, Newsome WT. Context-dependent computation by recurrent dynamics in prefrontal cortex. Nature. 2013;503:78–84.

3. Sussilo D, Churchland MM, Kaufman MT, Shenoy KV. A neural network that finds a naturalistic solution for the production of muscle activity. Nature Neuroscience. 2015;18(7):1025–1033.

4. Rajan K, Abbott LF. Recurrent network models of sequence generation and memory. Neuron. 2016;90:128–142.

5. Bolkan SS, Stujenske JM, Parnaudeau S, Spellman TJ, Rauffenbart C, Abbas AI, et al. Thalamic projections sustain prefrontal activity during working memory maintenance. Nature Neuroscience. 2017;20:987–996.

6. Goudar V, Buonomano DV. Encoding sensory and motor patterns as time-invariant trajectories in recurrent neural networks. eLife. 2018;7:e31134.

7. Song HF, Yang GR, Wang XJ. Training excitatory-inhibitory recurrent neural networks for cognitive tasks: a simple and flexible framework. PLoS Computational Biology. 2016;12(2):e1004792.

8. Yang GR, Joglekar MR, Song HF, Newsome WT, Wang XJ. Task representations in neural networks trained to perform many cognitive tasks. Nature Neuroscience. 2019;22:297–306.

9. Wolfgang M. Networks of spiking neurons: the third generation of neural network models. Neural Networks. 1997;10(9):1659–1671.

10. Ponulak F, Kasinski A. Supervised learning in spiking neural networks with ReSuMe: sequence learning, classification and spike-shifting. Neural Computation. 2010;22(2):467–510.

11. Sporea I, Grüning A. Supervised learning in multilayer spiking neural networks. Neural Computation. 2013;25(2):473–509.

12. Shrestha SB, Song Q. Adaptive learning rate of SpikeProp based on weight convergence analysis. Neural Networks. 2015;63:185–198.

13. Tavanaei A, Ghodrati M, Kheradpisheh SR, Masquelier T, Maida A. Deep learning in spiking neural networks. Neural Networks. 2019;111:47–63.

14. Panda P, Roy K. Learning to generate sequences with combination of Hebbian and non-Hebbian plasticity in recurrent spiking neural networks. Frontiers in Neuroscience. 2017;11:693.

15. Nicola W, Clopath C. Supervised learning in spiking neural networks with FORCE training. Nature Communications. 2017;8:2208.

16. Zenke F, Ganguli S. Superspike: Supervised learning in multilayer spiking neural networks. Neural Computation. 2018;30(6):1514–1541.

17. Neftci EO, Mostafa H, Zenke F. Surrogate gradient learning in spiking neural networks. IEEE Signal Processing Magazine. 2019;36:61–63.

18. Schmitt LI, Wimmer RD, Nakajima M, Happ M, Mofakham S, Halassa MM. Thalamic amplification of cortical connectivity sustains attentional control. Nature. 2017;545(7653):219–223.

19. Rikhye RV, Gilra A, Halassa MM. Thalamic regulation of switching between cortical representations enables cognitive flexibility. Nature Neuroscience. 2018;21:1753–1763.

20. Fujisawa S, Amarasingham A, Harrison MT, Buzsaki G. Behavior-dependent short-term assembly dynamics in the medial prefrontal cortex. Nature Neuroscience. 2008;11:823–833.

21. Harvey CD, Coen P, Tank DW. Choice-specific sequences in parietal cortex during a virtual-navigation decision task. Nature. 2012;484:62–68.

22. Hardy NF, Buonomano DV. Encoding time in feedforward trajectories of a recurrent neural network model. Neural Computation. 2018;30(2):378–396.

23. Ingrosso A, Abbott LF. Training dynamically balanced excitatory-inhibitory networks. PLoS ONE. 2019;14(8):e0220547.

24. Bellec G, Salaj D, Subramoney A, Legenstein R, Maass W. Long short-term memory and learning-to-learn in networks of spiking neurons. In: Advances in Neural Information Processing Systems (NIPS'18); 2018.

25. Bellec G, Scherr F, Subramoney A, Hajek E, Salaj D, Legenstein R, et al. A solution to the learning dilemma for recurrent networks of spiking neurons. Nature Communications. 2020;11:3625.

26. Okun M, Lampl I. Balance of excitation and inhibition. Scholarpedia. 2009;4(8):7467. doi:10.4249/scholarpedia.7467.

27. Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the 13th International Conference on Artificial Intelligence and Statistics; 2010. p. 249–256.

28. Rajan K, Abbott LF. Eigenvalue spectra of random matrices for neural networks. Physical Review Letters. 2006;97(18):188104.

29. Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:14126980. 2014;.

30. Orhan AE, Ma WJ. A diverse range of factors affect the nature of neural representations underlying short-term memory. Nature Neuroscience. 2019;22(2):275–283.

31. Kao JC. Considerations in using recurrent neural networks to probe neural dynamics. Journal of Neurophysiology. 2019;122:2504–2521.

32. van der Maaten LJP, Hinton GE. Visualizing data using t-SNE. Journal of Machine Learning Research. 2008;9:2579–2605.

33. Lundqvist M, Rose J, Herman P, Brincat SL, Buschman TJ, Miller EK. Gamma and beta bursts underlie working memory. Neuron. 2016;90:152–164.

34. Lundqvist M, Herman P, Miller EK. Working memory: delay activity, yes! persistent activity? maybe not. Journal of Neuroscience. 2018;38:7013–7019.

35. Miller EK, Lundqvist M, Bastos AM. Working memory 2.0. Neuron. 2018;100:463–475.

36. Zenke F, Poole B, Ganguli S. Continual learning through synaptic intelligence. In: Proceedings of International Conference on Machine Learning (ICML); 2017. p. 3978–3995.

37. Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, et al. Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences. 2017;114(13):3521–3526. doi:10.1073/pnas.1611835114.

38. Marton T, Seifikar H, Luongo FJ, Lee AT, Sohal VS. Roles of prefrontal cortex and mediodorsal thalamus in take engagement and behavioral flexibility. Journal of Neuroscience. 2018;38:2569–2578.

39. Lillicrap TP, Stantoro A, Marris L, Akerman CJ, Hinton GE. Backpropagation and the brain. Nature Review Neuroscience. 2020;21:335–346.

40. Song S, Abbott LF. Cortical development and remapping through spike timing-dependent plasticity. Neuron. 2001;32:339–350.

41. Fiete IR, Senn W, Wang CZH, Hahnloser RHR. Spike-time-dependent plasticity and heterosynaptic competition organize networks to produce long scale-free sequences of neural activity. Neuron. 2010;65:563–576.

42. Rezende DJ, Gerstner W. Stochastic variational learning in recurrent spiking networks. Frontiers in Computational Neuroscience. 2014;8:38.

43. Lee C, Panda P, Srinivasan G, Roy K. Training deep spiking convolutional neural networks with STDP-based unsupervised pre-training followed by supervised fine-tuning. Frontiers in Neuroscience. 2018;12:435. doi:10.3389/fnins.2018.00435.

44. Lobov SA, Mikhaylov AN, Shamshin M, Makarov VA, Kazantsev VB. Spatial properties of STDP in a self-learning spiking neural network enable controlling a mobile robot. Frontiers in Neuroscience. 2020;14:88. doi:10.3389/fnins.2020.00088.

45. Hao Y, Huang X, Dong M, Xu B. A biologically plausible supervised learning method for spiking neural networks using the symmetric STDP rule. Neural Networks. 2020;121:387–395.

46. Bush D, Philippides A, Husbands P, O'Shea M. Dual coding with STDP in a spiking recurrent neural network model of the hippocampus. PLoS Computational Biology. 2010;6(7):e1000839. doi:doi.org/10.1371/journal.pcbi.1000839.

47. Gillett M, Pereira U, Brunel N. Characteristics of sequential activity in networks with temporally asymmetric Hebbian learning. https://wwwbiorxivorg/content/101101/818773v1full. 2019;.

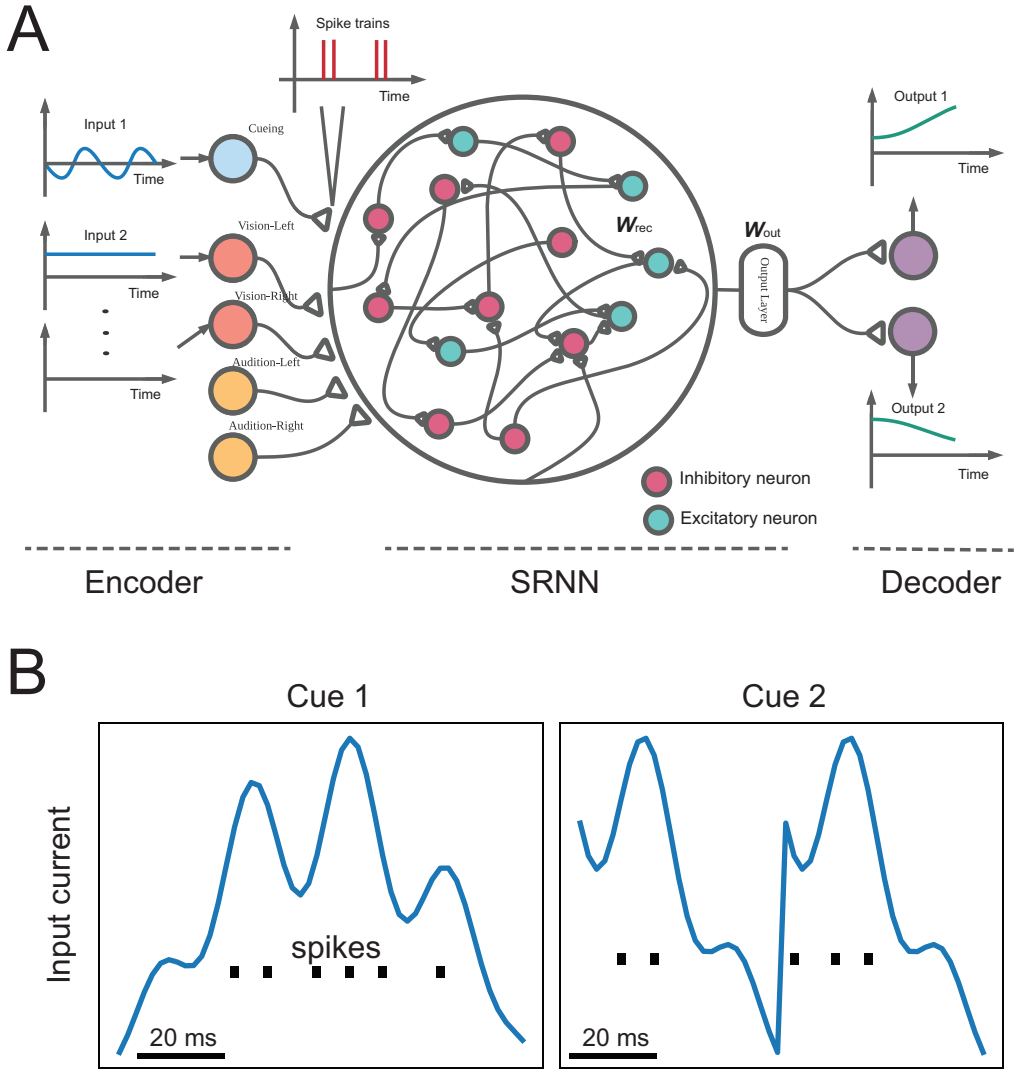

Fig 2. Schematic of computer simulations: an input encoder, a SRNN and an output decoder. (A) The SRNN consisting of both excitatory and inhibitory spiking neurons, receives the input spike trains converted from an encoder, and generates the output voltage and spike trains that are fed into a decoder, which generates the final decision. The neurons are fully connected with recurrent weights $\boldsymbol{W}_{\mathrm{rec}}$, and the readout layer was parameterized by $\boldsymbol{W}_{\mathrm{out}}$. (B) Illustration of encoding two sensory cues via spikes.

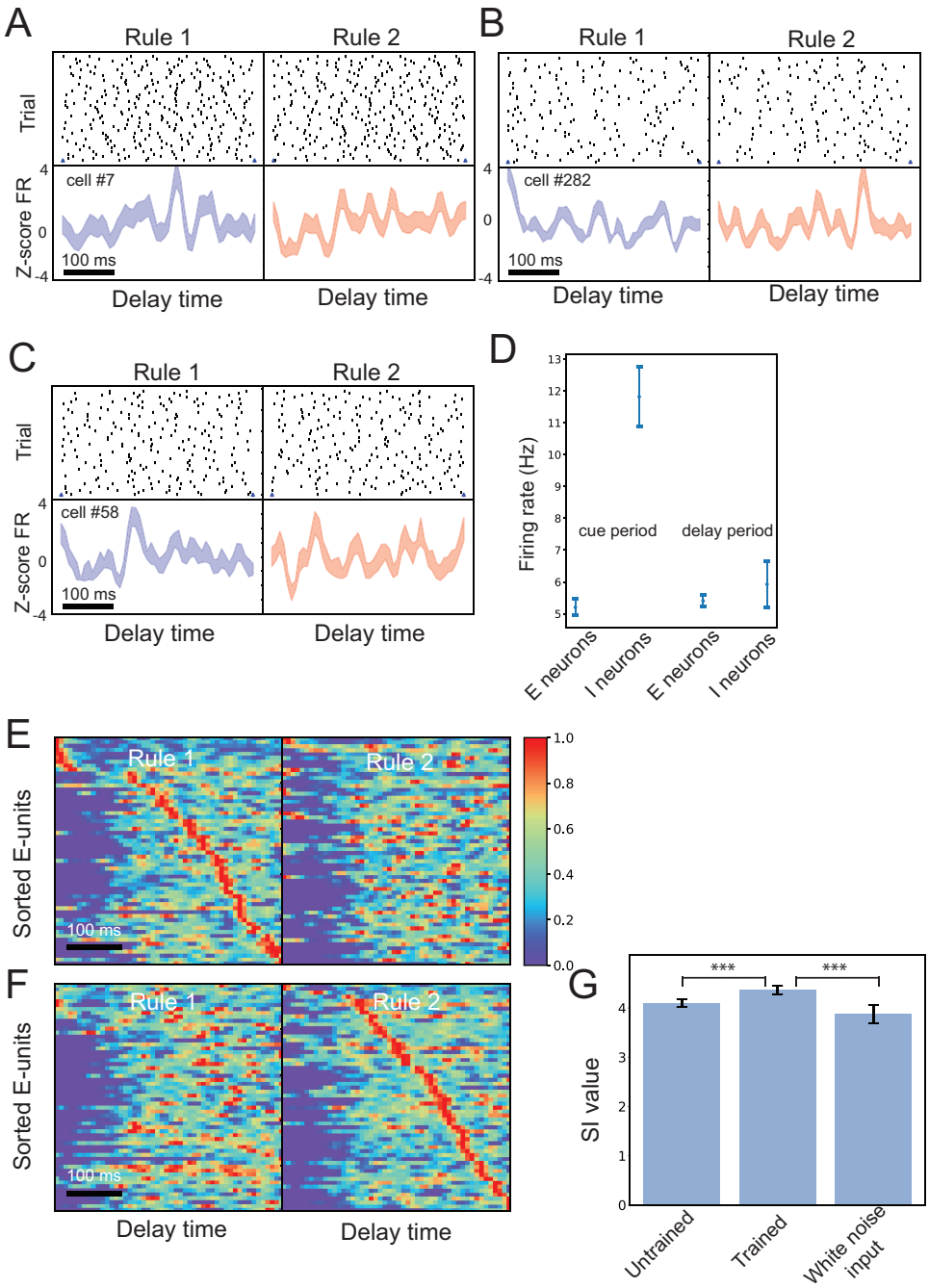

Fig 3. Rule-specific neural representations in trained SRNNs. (A-C) Representative spike rasters and PSTHs of two task-modulated neurons for rule representation. Panel A shows a representative neuron that has sharp tuning to rule 1, but not rule 2; panel B shows a representative neuron that has sharp tuning to rule 2, but not rule 1; panel C shows a representative neuron that has sharp tunings (but at different timings) with respect to two rules. Shaded area denotes SEM. (D) Statistics of averaged population firing rates during baseline (fixation) and delay periods. Error bar shows SEM across all neurons. (E,F) Heat maps (each row was normalized between 0 and 1) of 58 task-modulated excitatory neurons from one trained SRNN. Note that the sorted E-neurons formed a "neural sequence" for one rule, but not the other. In this example, sequentiality index (SI) was 4.52. (G) Comparison of the SI value for excitatory neurons under different conditions. Mean±SD statistics were computed from 10 untrained and 10 trained SRNNs. ***, $p < 10^{-3}$

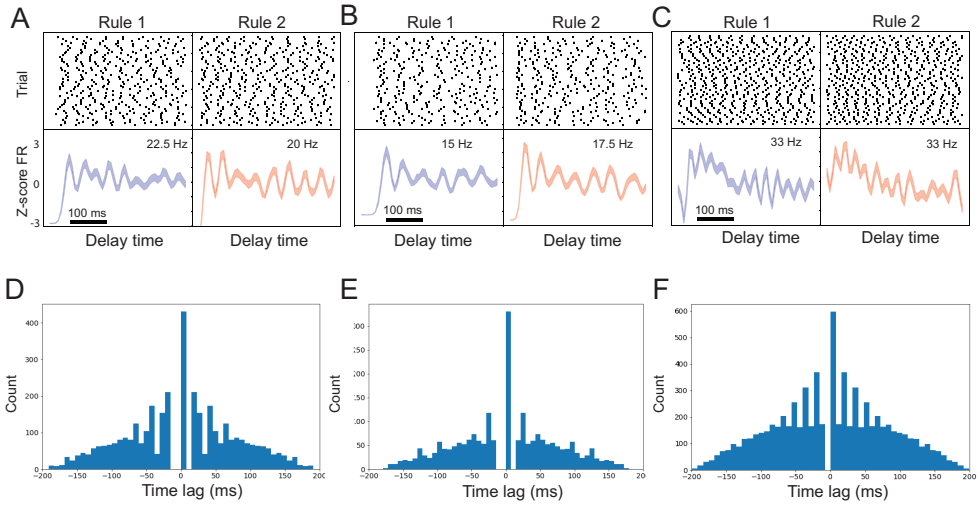

Fig 4. Spiking activities of inhibitory neurons of the trained SRNN showed beta or gamma oscillations. (A-C) Three examples of representative spike rasters and PSTHs that showed differential firing in frequency or phase between two rules. Error bar represents SEM. (D-F) Auto-correlograms of three neurons shown in panels A-C.

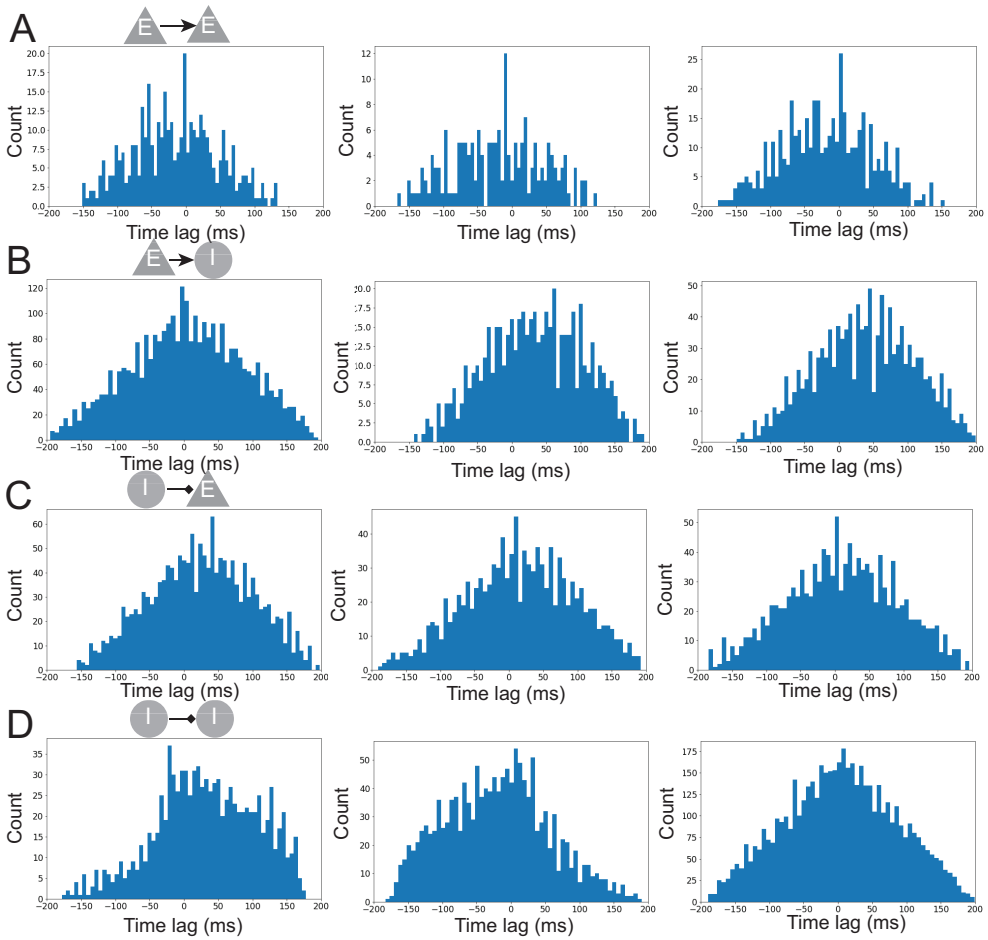

Fig 5. Coupling and cross-correlograms between rule-specific neurons. (A) Three cross-correlogram examples for E-E paired neurons. The trigger and target units are both rule-tuned neurons representing the same rule. (B) Three cross-correlogram examples of E-I paired neurons. The trigger unit is a rule-tuned neuron. (C) Three cross-correlogram examples of I-E paired neurons. The target unit is a rule-tuned neuron. (D) Three cross-correlogram examples of I-I paired neurons.

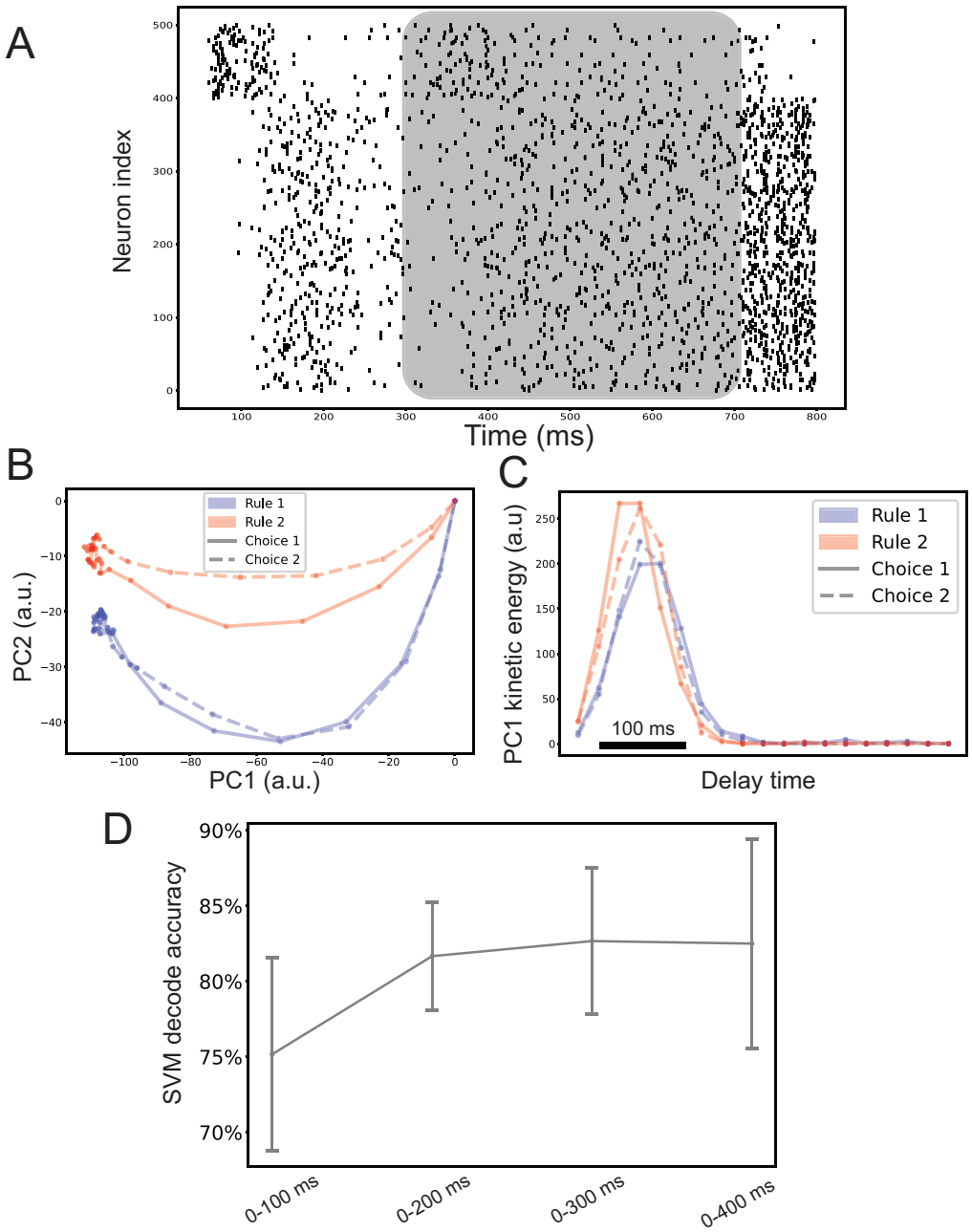

Fig 6. Neural trajectory analysis. (A) Spike rasters of 500 neurons during one single trial (neurons #1-400 are excitatory neurons). The shaded area denotes the delay period. (B) Neural trajectory $\mathbf{x}(t)$ of 400 excitatory neurons projected onto the PC1 and PC2 subspaces (from one trained SRNN) during the delay period. The rule-dependent information (blue vs. red) was maintained and propagated from the start (origin) to the end of delay period. Solid and dashed lines indicate the rule-dependent choices. (C) The curve of kinetic energy $\mathcal{K}(\mathbf{x})$ computed from PC1, which shows a high momentum in the first 100 ms, and then gradually decayed to very small value in the last 200 ms. (D) SVM decoding accuracy (mean±SD) based on two dominant PCs of 400 excitatory neuronal firing activity during the delay period.

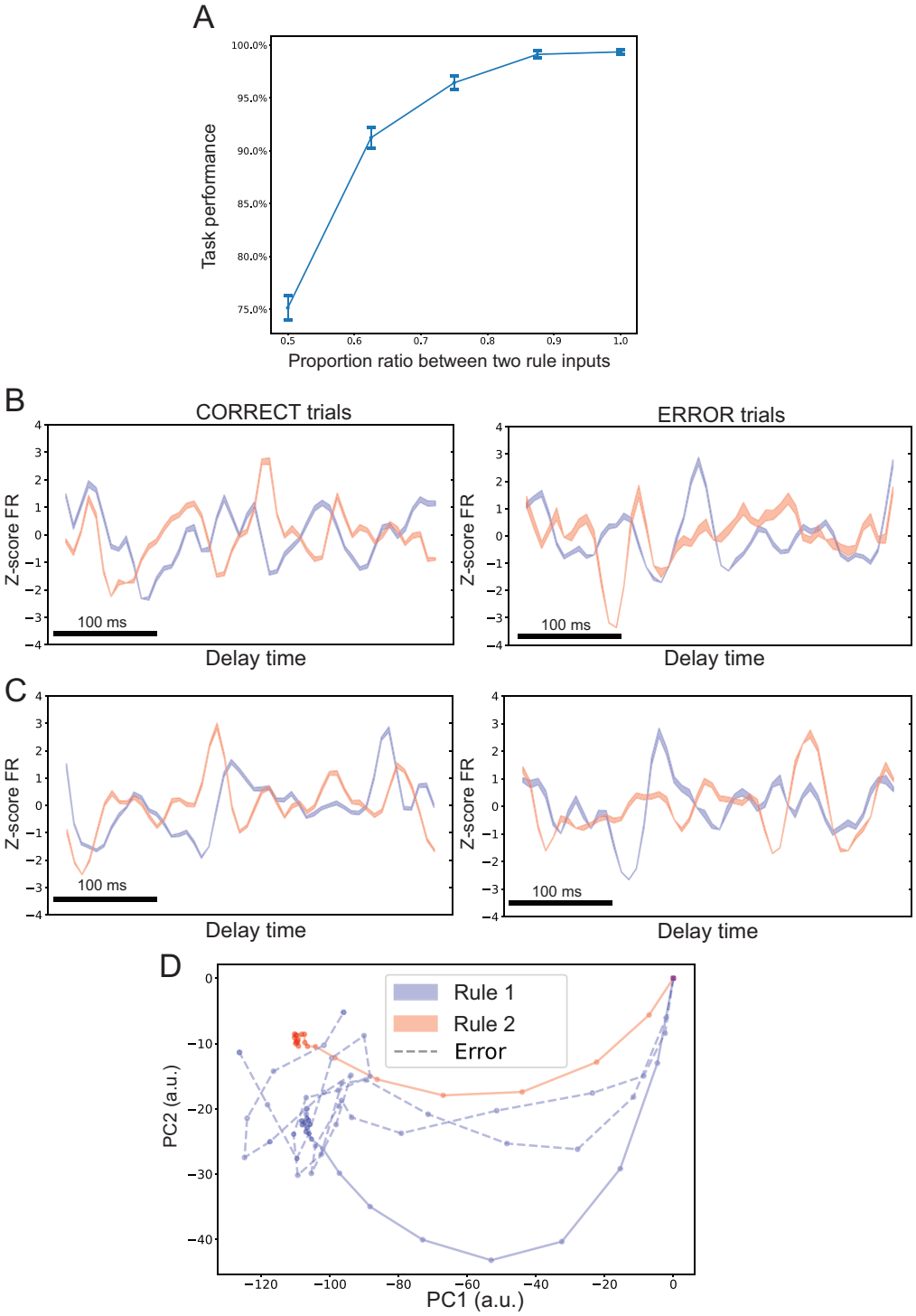

Fig 7. Error trials induced by rule-coding ambiguities. (A) Mean±SD performance curve of SRNN with respect to the mixed rule input with varying proportion ratios. (B,C) Examples of comparative PSTHs of two rule-tuned units between correct and error trials. (D) Projection of two representative error trials (dashed lines, during the delay period) onto the PC subspace yielded neural trajectories deviating from the correct trials. The error-trial trajectories representing rule 1 gradually deviated from the blue trajectory and wandered around at the end of the delay period, creating a higher kinetic energy during the last 200 ms.

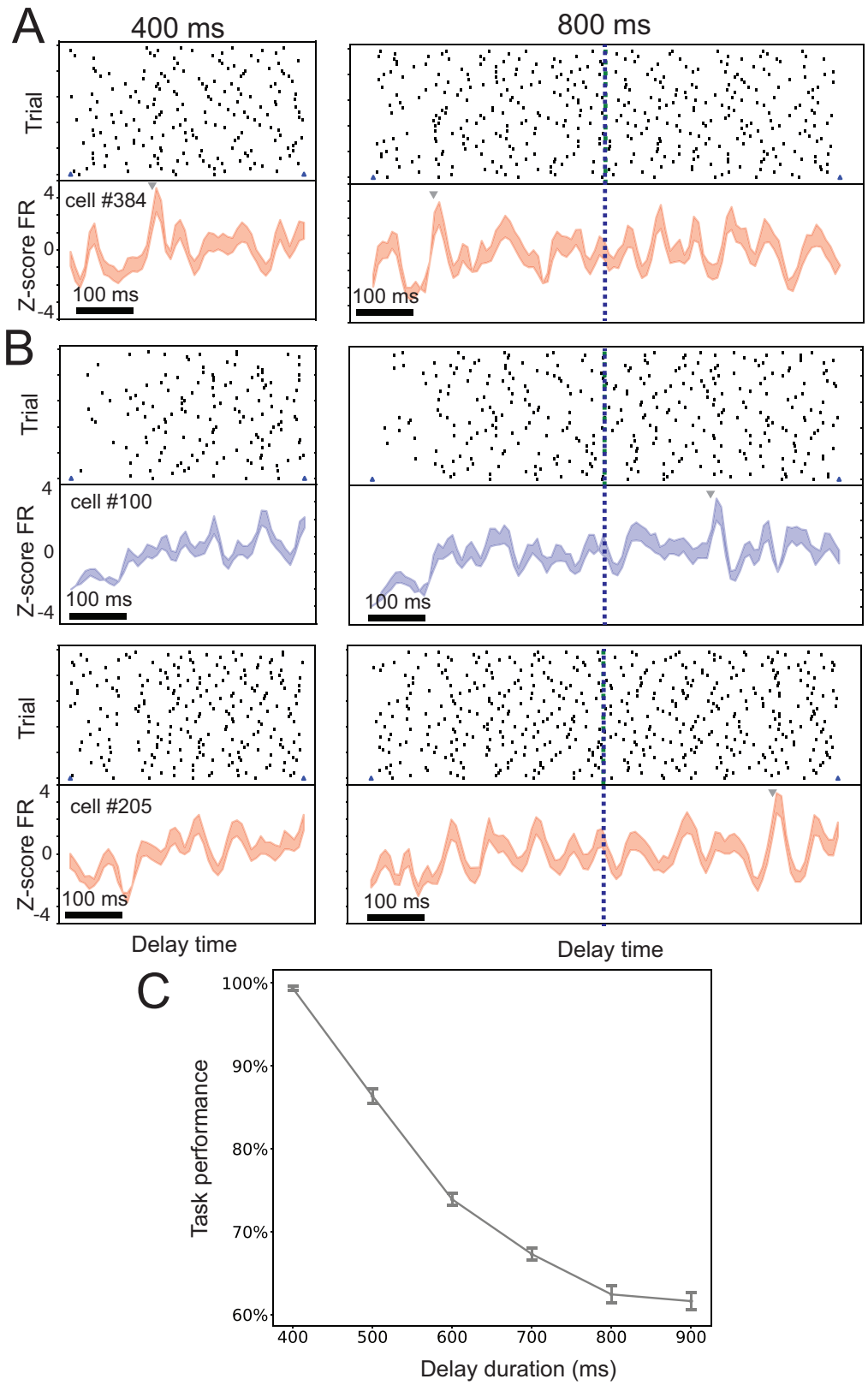

Fig 8. Neuronal activity during increased delay period. (A) One task-modulated neuron preserved the rule-specific peak timing in 800-ms as in 400-ms delay period. (B) New peak timing emerged in the late 800-ms delay period for two non-task-modulated neurons in the original setting. (C) Task performance degraded with an increased delay period.

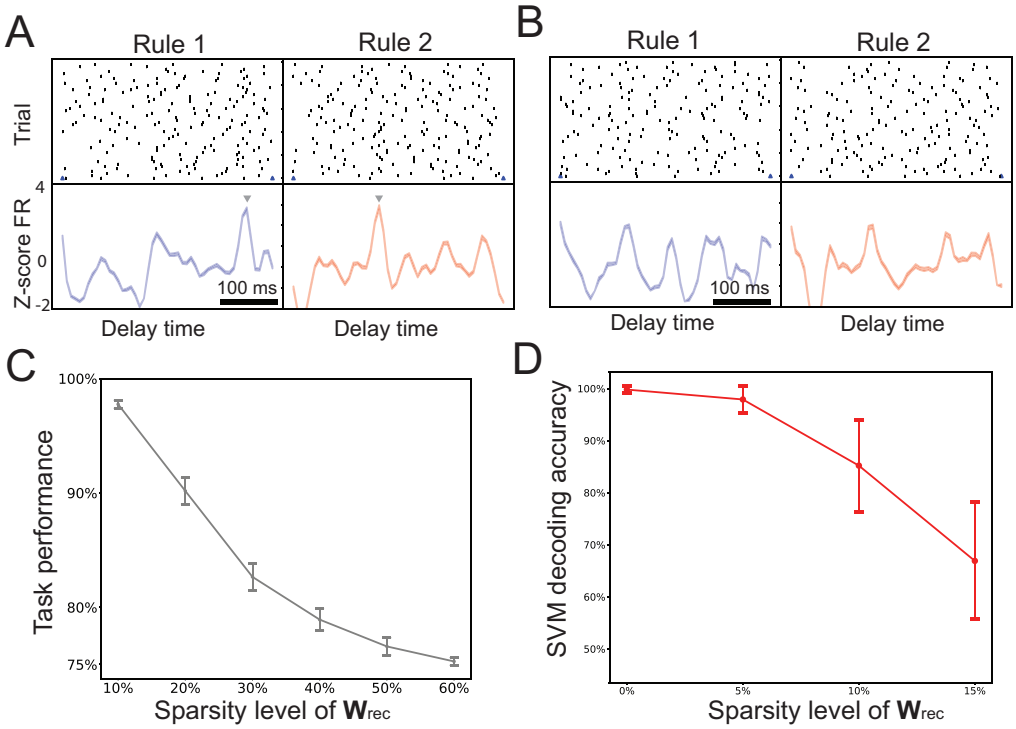

Fig 9. Impact of recurrent weight sparsification on neural representations and task performance. (A) Spike rasters and PSTHs of two task-modulated neurons during the delay period. Arrow marks the peak time. (B) The peak tuning of two neurons in panel A disappeared when $\boldsymbol{W}_{\text{rec}}$ was sparsified. (C) Task performance degraded as a result of recurrent weight sparsifiication. Error bar denotes SD across 10 random realizations. (D) The cross-validated SVM decoding accuracy degraded with an increasing degree of $\boldsymbol{W}_{\text{rec}}$ sparsity. Error bar denotes SD across 10 random realizations.

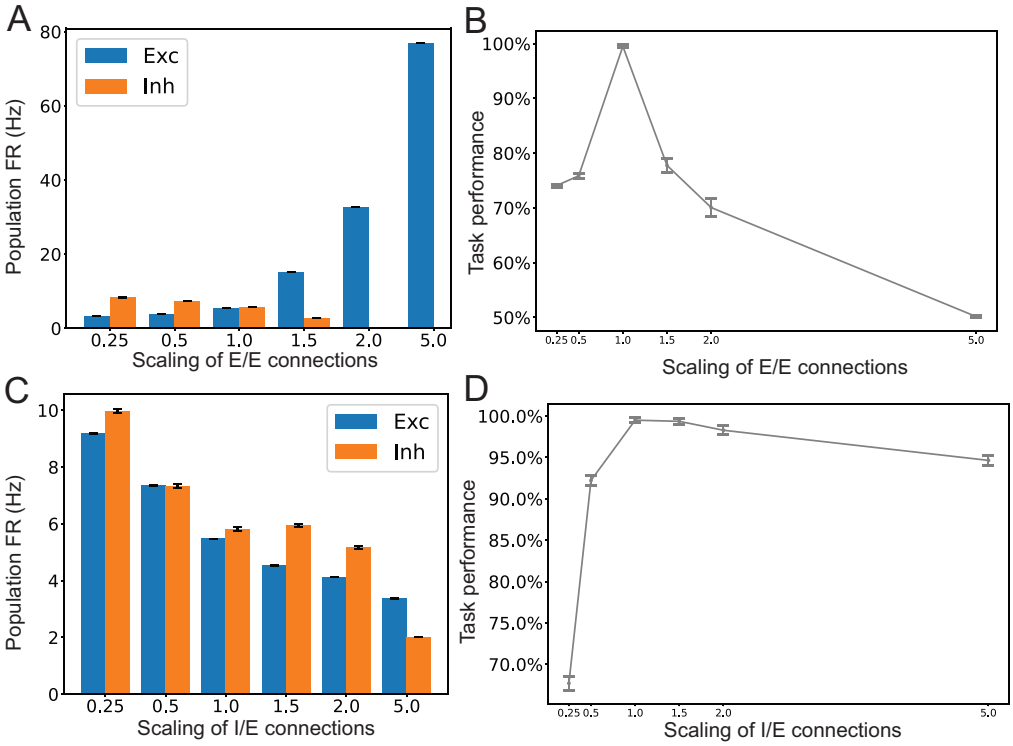

Fig 10. Impact of the cortical connectivity on the task performance and neural representations. (A) Population firing rates (FRs) of excitatory and inhibitory neurons changed with scaled EE connection strengths. Error bar denotes SEM. (B) Scaling up or down $\boldsymbol{W}_{\mathrm{EE}}$ changed the E/I level, and further affected the task performance. Error bar denotes the SD over 10 random realizations. (C,D) Similar to panels A and B, except for scaling IE connection strengths. (E) Peak firing rates of rule-tuned units decreased due to changed EE or IE connectivity.

Supporting Information

S1 Fig. Training SRNN. (**A**) Learning curve for the classification accuracy. (**B**) Learning curve for the regularized loss function. (**C**) Distributions of learned excitatory and inhibitory synaptic connections of a trained SRNN.

S2 Fig. Rule-specific neural sequences. Heat maps of mean firing rates (each row was normalized between 0 and 1) of 400 excitatory neurons from one trained SRNN.

S3 Fig. Visualization of functional clusters via stochastic neighbor embedding (tSNE). Red triangles denote excitatory neurons, and blue circles denote inhibitory neurons.

S4 Fig. Change in rule-specific tuning due to modified EE or IE connectivity. (A) Change of EE connectivity (scaling of 0.5) generated inappropriate tuning peaks at the opposite rule or reduced the peak tuning. (B) Change of IE connectivity (scaling of 1.5 and 2.0) changed or diminished the peak tuning.

S5 Fig. Rule-specific tuning and neural sequence were preserved during the delay period of the 4AFC task. (A) Spike rasters and PSTHs of two representative rule-tuned excitatory neurons. Shaded area denotes SEM. (B) Neural sequence formed by rule-specific excitatory neurons. The mean SI value was 4.55.

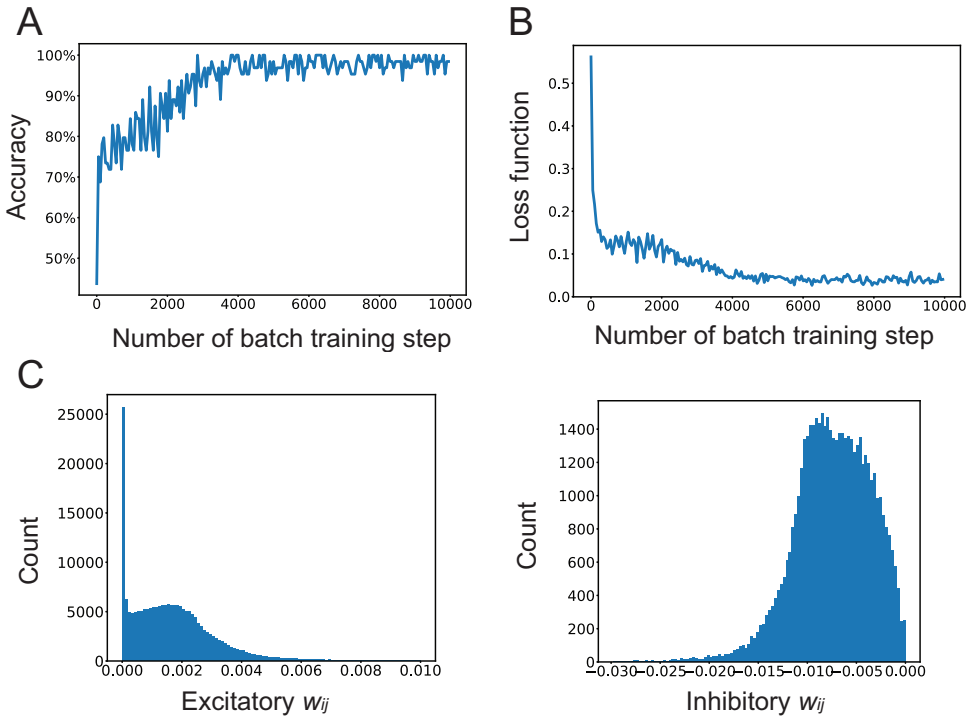

Fig S1. Training SRNN. (**A**) Learning curve for the classification accuracy. (**B**) Learning curve for the regularized loss function. (**C**) Distributions of learned excitatory and inhibitory synaptic connections of a trained SRNN.

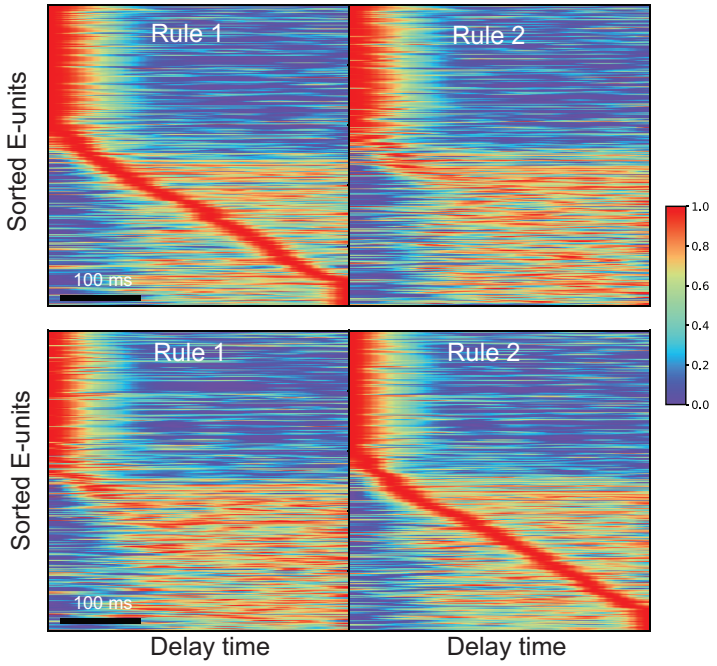

Fig S2. Rule-specific neural sequences. Heat maps of mean firing rates (each row was normalized between 0 and 1) of 400 excitatory neurons from one trained SRNN.

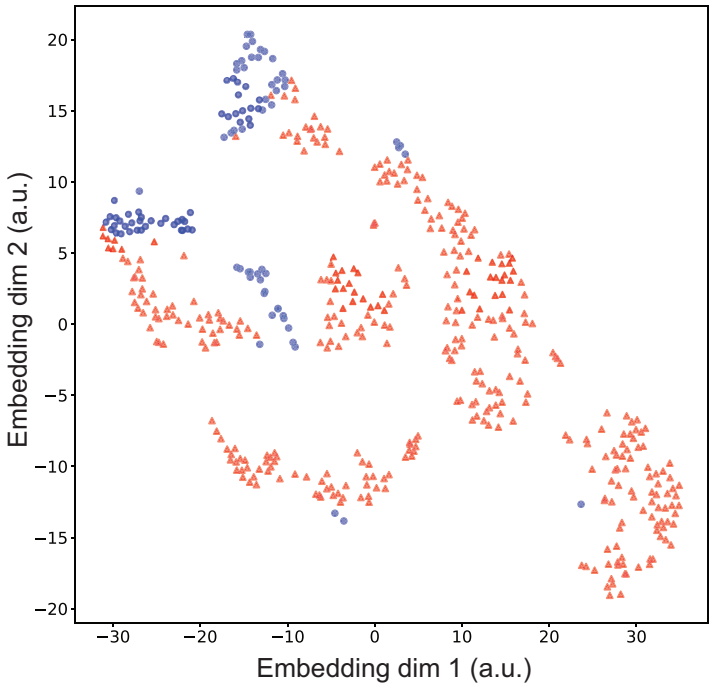

Fig S3. Visualization of functional clusters via stochastic neighbor embedding (tSNE). Red triangles denote excitatory neurons, and blue circles denote inhibitory neurons.

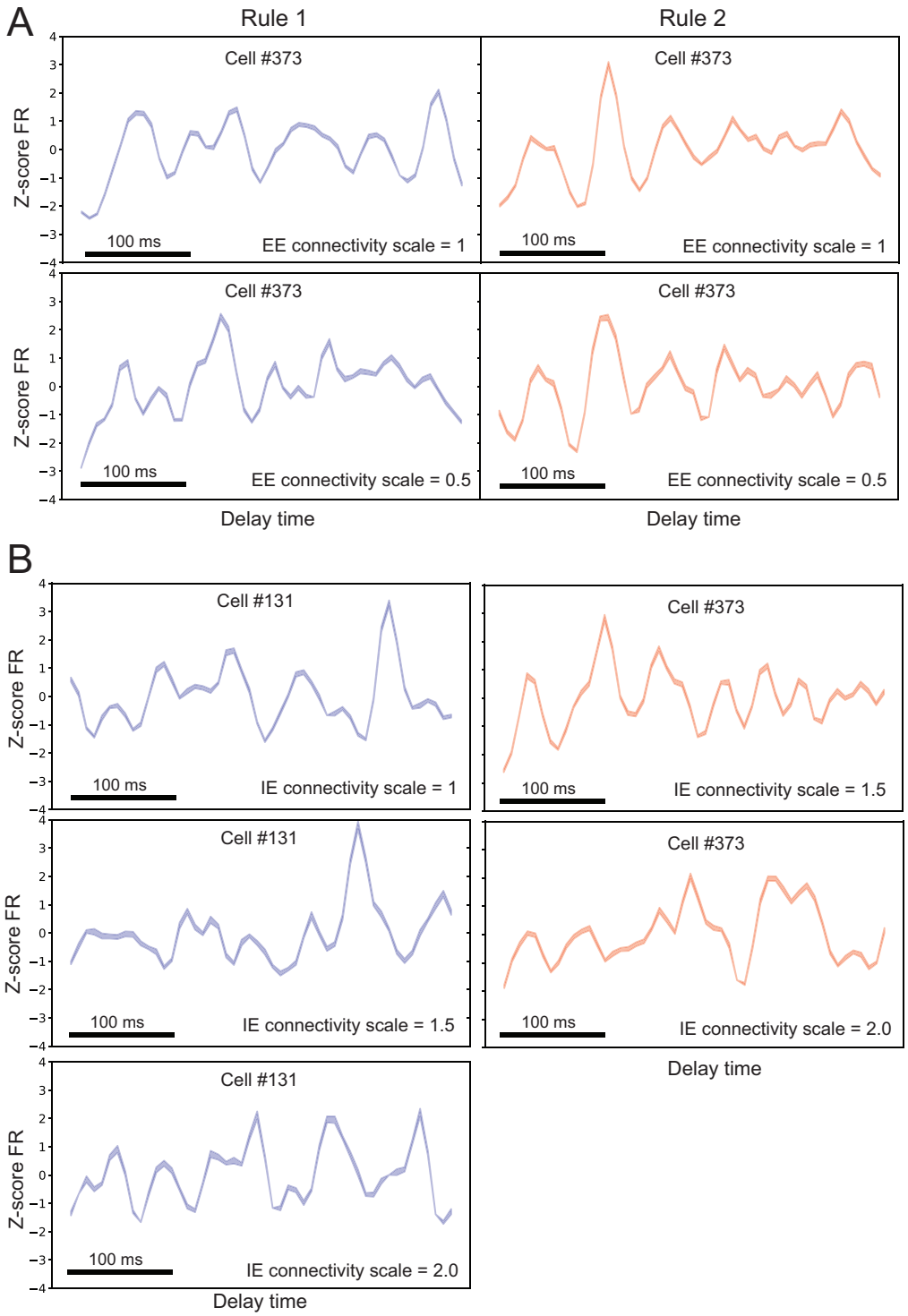

Fig S4. Change in rule-specific tuning due to modified EE or IE connectivity. (A) Change of EE connectivity (scaling of 0.5) generated inappropriate tuning peaks at the opposite rule or reduced the peak tuning. (B) Change of IE connectivity (scaling of 1.5 and 2.0) changed or diminished the peak tuning.

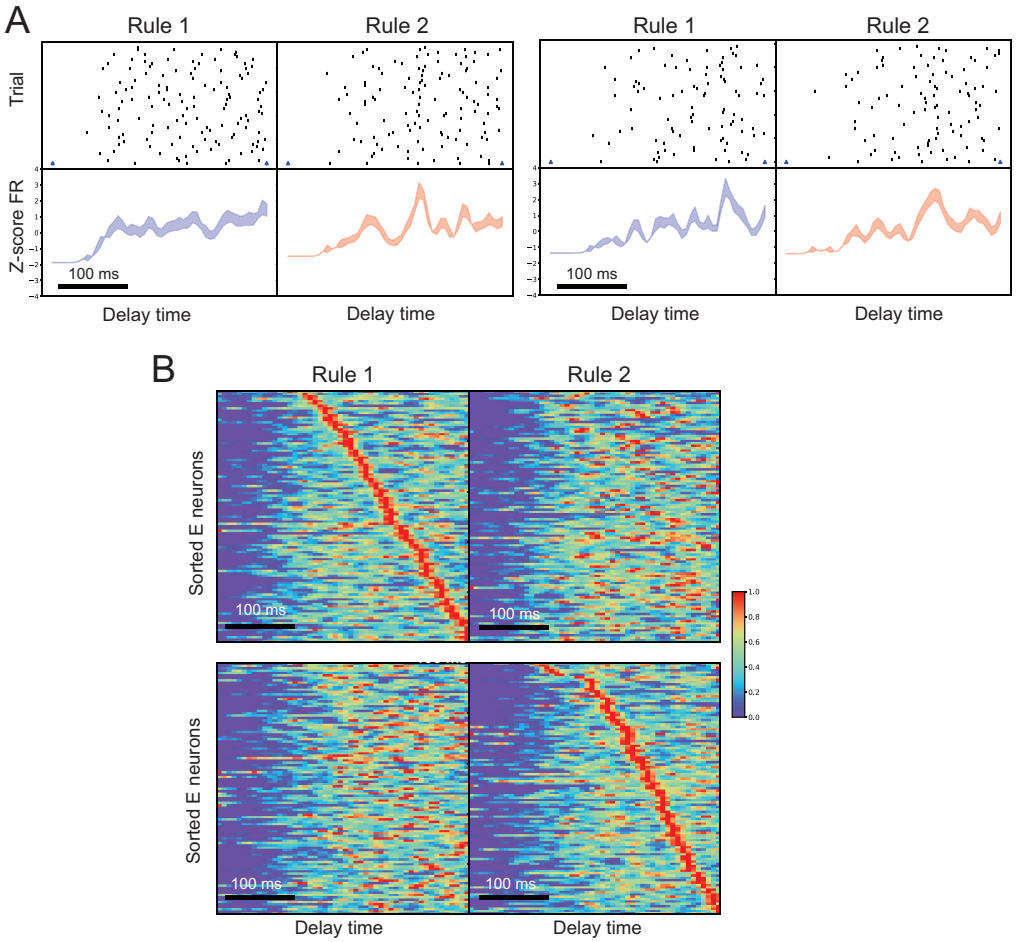

Fig S5. Rule-specific tuning and neural sequence were preserved during the delay period of the 4AFC task. (A) Spike rasters and PSTHs of two representative rule-tuned excitatory neurons. Shaded area denotes SEM. (B) Neural sequence formed by rule-specific excitatory neurons. The mean SI value was 4.55.