Simple framework for constructing functional spiking recurrent neural networks

Robert Kim\textsuperscript{a,b,c,1}, Yinghao Li\textsuperscript{a}, and Terrence J. Sejnowski\textsuperscript{a,d,e,1}

\textsuperscript{a}Computational Neurobiology Laboratory, Salk Institute for Biological Studies, La Jolla, CA 92037; \textsuperscript{b}Neurosciences Graduate Program, University of California San Diego, La Jolla, CA 92093; \textsuperscript{c}Medical Scientist Training Program, University of California San Diego, La Jolla, CA 92093; \textsuperscript{d}Institute for Neural Computation, University of California San Diego, La Jolla, CA 92093; and \textsuperscript{e}Division of Biological Sciences, University of California San Diego, La Jolla, CA 92093

Contributed by Terrence J. Sejnowski, September 5, 2019 (sent for review April 8, 2019; reviewed by Larry Abbott and David Sussillo)

Cortical microcircuits exhibit complex recurrent architectures that possess dynamically rich properties. The neurons that make up these microcircuits communicate mainly via discrete spikes, and it is not clear how spikes give rise to dynamics that can be used to perform computationally challenging tasks. In contrast, continuous models of rate-coding neurons can be trained to perform complex tasks. Here, we present a simple framework to construct biologically realistic spiking recurrent neural networks (RNNs) capable of learning a wide range of tasks. Our framework involves training a continuous-variable rate RNN with important biophysical constraints and transferring the learned dynamics to a spiking RNN in a one-to-one manner. The proposed framework introduces only 1 additional parameter to establish the equivalence between rate and spiking RNN models. We also study other model parameters related to the rate and spiking networks to optimize the one-to-one mapping. By establishing a close relationship between rate and spiking models, we demonstrate that spiking RNNs could be constructed to achieve similar performance as their counterpart continuous rate networks.

Significance

Recent advances in artificial intelligence and deep learning have significantly improved the capability of recurrently connected artificial neural networks. Although these networks can achieve high performance on various tasks, they often lack basic biophysical constraints, such as communication via spikes. However, recurrent microcircuitry in the brain can attain similar or better performance with discrete spikes in a much more efficient manner. Here, we introduce an extremely simple platform to construct spiking recurrent neural networks capable of performing numerous cognitive tasks commonly studied in neuroscience. Our method utilizes a close relationship between rate-based and spike-based networks that emerges under certain conditions. By characterizing these conditions, we provide another avenue that can be probed for constructing power-efficient spiking recurrent neural networks.

Author contributions: R.K. and T.J.S. designed research; R.K., Y.L., and T.J.S. performed research; R.K., Y.L., and T.J.S. analyzed data; and R.K., Y.L., and T.J.S. wrote the paper.

Reviewers: L.A., Columbia University; and D.S., Google.

The authors declare no competing interest.

Published under the PNAS license.

Data deposition: The data reported in this paper have been deposited in Open Science Framework, https://osf.io/4dj64/.

\textsuperscript{1}To whom correspondence may be addressed. Email: rkim@salk.edu or terry@salk.edu.

This article contains supporting information online at www.pnas.org/lookup/suppl/doi:10.1073/pnas.1905926116/-/DCSupplemental.

First published October 21, 2019.
the spiking RNNs in the same dynamic regime as their counterpart rate RNNs and takes advantage of the previously established methods to efficiently optimize network parameters while adhering to biophysical restrictions. These previously established methods include training a continuous-variable rate RNN using a gradient descent-based method (24–27) and connectivity weight matrix parametrization method to impose Dale’s principle (13). The gradient descent learning algorithm allowed us to easily optimize many parameters, including the connectivity weights of the network and the synaptic decay time constant for each unit. The weight parametrization method proposed by Song et al. (13) was utilized to enforce Dale’s principles and additional connectivity patterns without significantly affecting computational efficiency and network stability.

Combining these 2 existing methods with correct parameter values enabled us to directly map rate RNNs trained with backpropagation to LIF RNNs in a one-to-one manner. The parameters critical for mapping to succeed included the network size, the nonlinear activation function used for training rate RNNs, and a constant factor for scaling down the connectivity weights of the trained rate RNNs. Here, we investigated these parameters along with other LIF parameters and identified the range of values required for the mapping to be effective. We demonstrate that, when these parameters are set to their optimal values, the LIF models constructed from our framework can perform the same tasks that the rate models are trained to perform equally well.

**Results**

Here, we provide a brief overview of the 2 types of RNNs that we used throughout this study (more details are in Materials and Methods): continuous-variable firing rate RNNs and spiking RNNs. The continuous-variable-rate network model consisted of $N$ rate units with firing rates that were estimated via a nonlinear input–output transfer function (4, 5). The model was governed by the following set of equations:

$$
\tau_v \frac{dx_i}{dt} = -x_i + \sum_{j=1}^{N} w_{ij}^{rate} r_j^{rate} + I_{ext},
$$

where $\tau_v$ is the synaptic decay time constant for unit $i$, $x_i$ is the synaptic current variable for unit $i$, $w_{ij}^{rate}$ is the synaptic strength from unit $j$ to unit $i$, and $I_{ext}$ is the external current input to unit $i$. The firing rate of unit $i$ ($r_i^{rate}$) is given by applying a nonlinear transfer function $\phi(\cdot)$ to the synaptic current variable. Since the firing rates in spiking networks cannot be negative, we chose the activation function for our rate networks to be a nonnegative saturating function (standard sigmoid function) and parameterized the connectivity matrix ($w_{ij}^{rate} \in W^{rate}$) to enforce Dale’s principle and additional connectivity constraints (Materials and Methods).

The second RNN model that we considered was a network composed of $N$ spiking units. Throughout this study, we focused on networks of LIF units with membrane voltage dynamics that were given by

$$
\tau_m \frac{dv_i}{dt} = -v_i + \sum_{j=1}^{N} w_{ij}^{spk} v_j^{spk} + I_{ext},
$$

where $\tau_m$ is the membrane time constant (set to 10 ms throughout this study), $v_i$ is the membrane voltage of unit $i$, $w_{ij}^{spk}$ is the synaptic strength from unit $j$ to unit $i$, and $r_i^{rate}$ represents the synaptic filtering of the spike train of unit $j$, and $I_{ext}$ is the external current source. The discrete nature of $r_i^{spk}$ (Materials and Methods) poses a major challenge for directly training spiking networks using gradient-based supervised learning. Even though the main results presented here are based on LIF networks, our method can be generalized to quadratic integrate-and-fire (QIF) networks with only a few minor changes to the model parameters (SI Appendix, Table S1).

Continuous rate network training was implemented using the open source software library TensorFlow in Python, while LIF/QIF network simulations along with the rest of the analyses were performed in MATLAB.

**Training Continuous Rate Networks.** Throughout this study, we used a gradient-descent supervised method, known as backpropagation through time (BPTT), to train rate RNNs to produce target signals associated with a specific task (13, 24). The method that we used is similar to the one used by previous studies (13, 25, 27) (more details are in Materials and Methods) with 1 major difference in synaptic decay time constants. Instead of assigning a single time constant to be shared by all of the units in a network, our method tunes a synaptic constant for each unit using BPTT (Materials and Methods). Although tuning of synaptic time constants may not be biologically plausible, this feature was included to model diverse intrinsic synaptic timescales observed in single cortical neurons (28–30).

We trained rate RNNs of various sizes on a simple task modeled after a Go-NoGo task to demonstrate our training method (Fig. 1). Each network was trained to produce a positive mean population activity approaching +1 after a brief input pulse (Fig. 1A). For a trial without an input pulse (i.e., NoGo trial), the networks were trained to maintain the output signal close to 0. The units in a rate RNN were sparsely connected via $W^{rate}$ and received a task-specific input signal through weights ($W^{out}$) drawn from a normal distribution with 0 mean and unit variance. The network output ($o^{rate}$) was then computed using a set of linear readout weights:

$$
o^{rate}(t) = W^{rate} o^{out}(t),
$$

where $W^{rate}$ is the readout weights and $o^{out}(t)$ is the firing rate estimates from all of the units in the network at time $t$. The recurrent weight matrix ($W^{rate}$), the readout weights ($W^{out}$), and the synaptic decay time constants ($\tau_v$) were optimized during training, while the input weight matrix ($W^{in}$) stayed fixed (Materials and Methods).

The network size ($N$) was varied from 10 to 400 (9 different sizes), and 100 networks with random initializations were trained for each size. For all of the networks, the minimum and maximum synaptic decay time constants were fixed to 20 and 50 ms, respectively. As expected, the smallest rate RNNs ($N = 10$) took the longest to train, and only 69% of the rate networks with $N = 10$ were successfully trained (Fig. 1C; SI Appendix has training termination criteria).

**One-to-One Mapping from Continuous Rate Networks to Spiking Networks.** We developed a simple procedure that directly maps dynamics of a trained continuous rate RNN to a spiking RNN in a one-to-one manner.

In our framework, the 3 sets of the weight matrices ($W^{in}$, $W^{rate}$, and $W^{out}$) along with the tuned synaptic time constants ($\tau_v$) from a trained rate RNN are transferred to a network of LIF spiking units. The spiking RNN is initialized to have the same topology as the rate RNN. The input weight matrix and the synaptic time constants are simply transferred without any modification, but the recurrent connectivity and the readout weights need to be scaled by a constant factor ($\lambda$) in order to account
for the difference in the firing rate scales between the rate model and the spiking model (Materials and Methods and Fig. 2A). The effects of the scaling factor are clear in an example LIF RNN model constructed from a rate model trained to perform the Go-NoGo task. The rate RNN model contained excitatory (red circles) and inhibitory (blue circles) units. (B) Distribution of the tuned synaptic decay time constants (mean ± SD, 28.2 ± 9.4 ms; Left) and the average trained rate RNN task performance (Right) from an example rate RNN model. The mean ± SD output signals from 50 Go trials (dark purple) and from 50 NoGo trials (light purple) are shown. The green box represents the input stimulus given for the Go trials. The rate RNN contained 200 units (169 excitatory and 31 inhibitory units). (C) Rate RNNs with different network sizes trained to perform the Go-NoGo task. For each network size, 100 RNNs with random initial conditions were trained. All of the networks successfully trained performed the task almost perfectly (range from 96 to 100%; Left). As the network size increased, the number of training trials decreased (mean ± SD is shown; Right).

Fig. 1. Rate RNNs trained to perform the Go-NoGo task. (A) Schematic diagram illustrating a continuous rate RNN model trained to perform the Go-NoGo task. The rate RNN model contained excitatory (red circles) and inhibitory (blue circles) units. (B) Distribution of the tuned synaptic decay time constants (mean ± SD, 28.2 ± 9.4 ms; Left) and the average trained rate RNN task performance (Right) from an example rate RNN model. The mean ± SD output signals from 50 Go trials (dark purple) and from 50 NoGo trials (light purple) are shown. The green box represents the input stimulus given for the Go trials. The rate RNN contained 200 units (169 excitatory and 31 inhibitory units). (C) Rate RNNs with different network sizes trained to perform the Go-NoGo task. For each network size, 100 RNNs with random initial conditions were trained. All of the networks successfully trained performed the task almost perfectly (range from 96 to 100%; Left). As the network size increased, the number of training trials decreased (mean ± SD is shown; Right).

LIF Networks for Context-Dependent Input Integration. The Go-NoGo task considered in the previous section did not require complex cognitive computations. In this section, we consider a more complex task and probe whether spiking RNNs can be constructed from trained rate networks in a similar fashion. The task considered here is modeled after the context-dependent sensory integration task used by Mante et al. (7). Briefly, Mante et al. (7) trained rhesus monkeys to integrate inputs from one sensory modality (dominant color or dominant motion of randomly moving dots) while ignoring inputs from the other modality (7). A contextual cue was also given to instruct the monkeys which sensory modality they should attend to. The task required the monkeys to utilize flexible computations, as the same modality can be either relevant or irrelevant depending on the contextual cue. Previous works have successfully trained continuous rate RNNs to perform a simplified version of the task and replicated the neural dynamics present in the experimental data (7, 13, 15). Using our framework, we constructed a spiking RNN model that can perform the task and capture the dynamics observed in the experimental data.

For the task paradigm, we adopted a similar design as the one used by the previous modeling studies (7, 13, 15). A network of recurrently connected units received 2 streams of noisy input signals along with a constant-valued signal that encoded the contextual cue (Materials and Methods and Fig. 3d). To simulate a noisy sensory input signal, a random Gaussian time series signal with 0 mean and unit variance was first generated. Each input signal was then shifted by a positive or negative constant (“offset”) to encode evidence toward the (+) or (−) choice, respectively. Therefore, the offset value determined how much evidence for the specific choice was represented in the noisy input signal. The network was trained to produce an output signal approaching
LIF RNNs

Rate RNNs

Input

Output

Fig. 2. Mapping trained rate RNNs to LIF RNNs for the Go-NoGo task. (A) Schematic diagram illustrating direct mapping from a continuous rate RNN model (Upper) to a spiking RNN model (Lower). The optimized synaptic decay time constants ($\tau_{\text{in}}$) along with the weight parameters ($W_{\text{in}}$, $W_{\text{rate}}$, and $W_{\text{out}}$) were transferred to a spiking network with LIF units (red and blue circles with dashed outlines). The connectivity and the readout weights were scaled by a constant factor, $\lambda$. (B) LIF RNN performance on the Go-NoGo task without scaling ($\lambda = 1$; Left), with insufficient scaling (Center), and with appropriate scaling (Right). The network contained 200 units (169 excitatory and 31 inhibitory units). Mean ± SD over 50 Go and 50 NoGo trials. (C) Successfully converted LIF networks and their average task performance on the Go-NoGo task with different network sizes. All of the rate RNNs trained in Fig. 1 were converted to LIF RNNs. The network size was varied from $N = 10$ to 400. (D) Average synaptic decay values for $N = 250$ across different maximum synaptic decay constants. (E) Successfully converted LIF RNNs and their average task performance on the Go-NoGo task with fixed network size ($N = 250$) and different maximum synaptic decay constants. The maximum synaptic decay constants were varied from 20 to 1,000 ms.

+1 (or −1) if the cued input signal had a positive (or negative) mean. For example, if the cued input signal was generated using a positive offset value, then the network should produce an output that approaches +1 regardless of the mean of the irrelevant input signal.

Rate networks with different sizes ($N = 10, 50, \ldots, 450, 500$) were trained to perform the task. As this is a more complex task compared with the Go-NoGo task considered in the previous section, the numbers of units and trials required to train rate RNNs were larger than those in the models trained on the Go-NoGo task (Fig. 3 B and C). The synaptic decay time constants were again limited to a range of 20 and 50 ms, and 100 rate RNNs with random initial conditions were trained for each network size. For the smallest network size ($N = 10$), the rate networks could not be trained to perform the task within the first 6,000 trials (Fig. 3B).

Next, all of the rate networks successfully trained for the task were transformed into LIF models. Example output responses along with the distribution of the tuned synaptic decay constants from a converted LIF model ($N = 250$, $\tau_{\text{min}} = 20$ ms, $\tau_{\text{max}} = 50$ ms) are shown in Fig. 4 A and B. The task performance of the LIF model was 98% and comparable with the rate RNN used to construct the spiking model (Fig. 4C). In addition, the LIF network manifested population dynamics similar to the dynamics observed in the group of neurons recorded by Mante et al. (7) and rate RNN models investigated in previous studies (7, 13, 15): individual LIF units displayed mixed representation of the 4 task variables (modality 1, modality 2, network choice, and context) (SI Appendix, Fig. S5A), and the network revealed the characteristic line attractor dynamics (SI Appendix, Fig. S5B).

Similar to the spiking networks constructed for the Go-NoGo task, the LIF RNNs performed the input integration task more accurately as the network size increased (Fig. 4D). Next, the network size was fixed to $N = 250$, and $\tau_{\text{min}}$ was gradually increased from 20 to 1,000 ms. For $\tau_{\text{min}} = 50$ ms, all 100 rate networks failed to learn the task within the first 6,000 trials. The conversion from the rate models to the LIF models did not lead to significant loss in task performance for all of the other maximum decay constant values considered (Fig. 4E).

Analysis of the Conversion Method. Previous sections illustrated that our framework for converting rate RNNs to LIF RNNs is robust as long as the network size is not too small ($N \geq 200$), and the optimal size was $N = 250$ for both tasks. When the network size is too small, it is harder to train rate RNNs, and the rate models successfully trained do not reliably translate to spiking networks (Figs. 2D and 4D). In this section, we further investigate the relationship between rate and LIF RNN models and characterize other parameters crucial for the conversion to be effective.

Training synaptic decay time constants. As shown in Fig. 5, training the synaptic decay constants for all of the rate units is not required for the conversion to work. Rate RNNs (100 models with different initial conditions) with the synaptic decay time constant fixed to 35 ms (average $\tau^+$ value for the networks trained with $\tau_{\text{min}} = 20$ ms and $\tau_{\text{max}} = 50$ ms) were trained on the Go-NoGo task and converted to LIF RNNs (Fig. 5). The task performance of these LIF networks was not significantly different from the performance of the spiking models with optimized synaptic decay constants bounded between 20 and 50 ms. The number of the successful LIF models with
the fixed synaptic decay constant was also comparable with the number of the successful LIF models with the tuned decay constants (Fig. 5).

**Other LIF parameters.** We also probed how LIF model parameters affected our framework. More specifically, we focused on the refractory period and synaptic filtering. The LIF models constructed in the previous sections used an absolute refractory period of 2 ms and a double-exponential synaptic filter (Materials and Methods). Rate models ($N = 250$ and $\tau_{\text{max}} = 100$ ms) trained on the sensory integration task were converted to LIF networks with different values of the refractory period. As the refractory period became longer, the task performance of the spiking RNNs decreased rapidly (Fig. 6A). When the refractory period was set to 0 ms, the LIF RNNs still performed the integration task with a moderately high average accuracy (92.8 ± 14.3%), but the best task performance was achieved when the refractory period was set to 2 ms (average performance, 97.0 ± 6.6%) (Fig. 6A, Inset).

We also investigated how different synaptic filters influenced the mapping process. We first fixed the refractory period to its optimal value (2 ms) and constructed 100 LIF networks ($N = 250$) for the integration task using a double synaptic filter (Materials and Methods and Fig. 6B, light blue). Next, the synaptic filter was changed to the following single-exponential filter:

$$r_i^{\text{d}} \frac{dr_i^{\text{spk}}}{dt} = -r_i^{\text{spk}} + \sum_{t' < t} \delta(t - t_i^{\text{spk}}),$$

where $r_i^{\text{spk}}$ represents the filtered spike train of unit $i$ and $t_i^{\text{spk}}$ refers to the $k$th spike emitted by unit $i$. The task performance of the LIF networks with the above single-exponential synaptic filter was 95.7 ± 7.3%, and it was not significantly different from the performance of the double-exponential synaptic LIF models (97.0 ± 6.6%) (Fig. 6B).

**Initial connectivity weight scaling.** We considered the role of the connectivity weight initialization in our framework. In the previous sections, the connectivity weights ($W_{\text{init}}$) of the rate networks were initialized as random, sparse matrices with 0 mean and an SD of $g/\sqrt{N \cdot P}$, where $g = 1.5$ is the gain term that controls the dynamic regime of the networks and $P = 0.20$ is the initial connectivity probability (Materials and Methods). Previous studies have shown that rate networks operating in a high gain regime ($g > 1.0$) produce chaotic spontaneous trajectories, and this rich dynamics can be harnessed to perform complex computations (6, 11). By varying the gain term, we determined if highly chaotic initial dynamics were required for successful conversion. We considered 6 different gain terms ranging from 0.5 to 3.5, and for each gain term, we constructed 100 LIF RNNs (from 100 rate RNNs with random initial conditions) (Fig. 6C) to perform the contextual integration task. The LIF models performed the task equally well across all of the gain terms considered (no statistical significance detected).

**Transfer function.** One of the most important factors that determines whether rate RNNs can be mapped to LIF RNNs in a one-to-one manner is the nonlinear transfer function used in the rate models. We considered 3 nonnegative transfer functions commonly used in the machine learning field to train rate RNNs on the Go-NoGo task: sigmoid, rectified linear, and softplus functions (Fig. 7A and SI Appendix). For each transfer function, 100 rate models ($N = 250$ and $\tau_{\text{max}} = 50$ ms) were trained. Although all 300 rate models were trained to perform the task almost perfectly (Fig. 7B), the average task performance and the number of successful LIF RNNs were highest for the rate models trained with the sigmoid transfer function (Fig. 7C). None of the rate models trained with the rectified linear transfer function could be successfully mapped to LIF models, while the spiking networks constructed from the rate models trained with the softplus function were not robust and produced incorrect responses (SI Appendix, Fig. S6).
Discussion
In this study, we presented a simple framework that harnesses the dynamics of trained continuous rate models to produce functional spiking RNN models. We identified a set of parameters required to directly transform trained rate RNNs to LIF models, thus establishing a one-to-one correspondence between these 2 model types. Despite of additional spiking-related parameters, surprisingly only a single parameter (i.e., scaling factor) was required for LIF RNN models to closely mimic their counterpart rate models. Furthermore, this framework can flexibly impose functional connectivity constraints and heterogeneous synaptic time constants.

We investigated and characterized the effects of several model parameters on the stability of the transfer learning from rate models to spiking models. The parameters critical for the mapping to be robust included the network size, choice of activation function for training rate RNNs, and a constant factor to scale down the connectivity weights of the trained rate networks. Although the softplus and rectified linear activation functions are popular for training deep neural networks, we demonstrated that a constant factor to scale rate RNNs trained in Fig. 2 was not significantly different from the performance of the LIF RNNs with tuned synaptic decay time constants (Fig. 6A). Training the synaptic decay time constants, choice of synaptic filter (between single- and double-exponential filter),

Another important parameter was the constant scaling factor used to scale $W^\text{rate}$ and $W^\text{out}$ before transferring them to LIF networks. When the scaling factor was set to its optimal value (found via grid search), the LIF units behaved like their counterpart rate units, and the spiking networks performed the tasks that the rate RNNs were trained to perform (Fig. 2). Another parameter that affected the reliability of the conversion was the refractory period parameter of the LIF network models. The LIF performance was optimal when the refractory was set to 2 ms (Fig. 6A). Training the synaptic decay time constants, choice of synaptic filter (between single- and double-exponential filter),

![Image](https://www.pnas.org/content/116/24/116)

Fig. 4. LIF network models constructed to perform the contextual integration task. (A) Example output responses and spike raster plots from an LIF network model for 2 different input stimuli (rows) and 2 contexts (columns). The network contained 250 units (188 excitatory and 62 inhibitory units), and the noisy input signals were scaled by 0.5 vertically for better visualization of the network responses (purple lines). (B) Distribution of the optimized synaptic decay time constants ($\tau^d$) for the example LIF network (mean ± SD, 38.9 ± 9.3 ms). The time constants were limited to range between 20 and 50 ms. (C) Average output responses of the example LIF network. Mean ± SD network responses across 100 randomly generated trials are shown. (D) Successfully converted LIF networks and their average task performance across different network sizes. The network size was varied from $N = 10$ to 500. The rate RNNs trained in Fig. 3 were used. (E) Successfully converted LIF networks with $N = 250$ and their average task performance across different maximum synaptic decay constants (varied from 20 to 1,000 ms).

![Image](https://www.pnas.org/content/116/24/116)

Fig. 5. Optimizing synaptic decay constants is not required for conversion of rate RNNs. The Go-NoGo task performance of the LIF RNNs constructed from the rate networks with a fixed synaptic constant ($\tau^d = 35$ ms; blue) was not significantly different from the performance of the LIF RNNs with tuned synaptic decay time constants ($\tau^d_{\text{min}} = 20$ ms, $\tau^d_{\text{max}} = 50$ ms; green).
and connectivity weight initialization did not affect the mapping procedure (Figs. 5 and 6 B and C).

The type of approach used in this study (i.e., conversion of a rate network to a spiking network) has been previously used in neuromorphic engineering to construct power-efficient deep spiking networks (31–36). These studies mainly used feedforward multilayer networks or convolutional neural networks aimed to accurately classify input signals or images without placing too much emphasis on biophysical limitations. The overarching goal in these studies was to maximize task performance while minimizing power consumption and computational cost. However, the main aim of this study was to construct spiking recurrent network models that abide by important biological constraints in order to relate emerging mechanisms and dynamics to experimentally observed findings. To this end, we have carefully designed our continuous rate RNNs to include several biological features. These include 1) recurrent architectures, 2) sparse connectivity that respects Dale’s principle, and 3) heterogeneous synaptic decay time constants.

For constructing spiking RNNs, recent studies have proposed methods that built on the FORCE method to train spiking RNNs (8, 20–22). Conceptually, our work is most similar to the work by DePasquale et al. (21). The method developed by DePasquale et al. (21) also relies on mapping a trained continuous-variable rate RNN to a spiking RNN model. However, the rate RNN model used in their study was designed to provide dynamically rich auxiliary basis functions meant to be distributed to overlapping populations of spiking units. Due to this reason, the relationship between their rate and spiking models is rather complex, and it is not straightforward to impose functional connectivity constraints on their spiking RNN model. An additional procedure was introduced to implement Dale’s principle, but this led to more fragile spiking networks with considerably increased training time (21). The one-to-one mapping between rate and spiking networks used in our method solved these problems without sacrificing network stability and computational cost: biophysical constraints that we wanted to incorporate into our spiking model were implemented in our rate network model first and then, transferred to the spiking model.

While our framework incorporated the basic yet important biological constraints, there are several features that are also not biologically realistic in our models. The gradient-descent method used to tune the rate model parameters, including the connectivity weights and the synaptic decay time constants, in a supervised manner is not biologically plausible. Although tuning of the synaptic time constants is not realistic and has not been observed experimentally, previous studies have underscored the importance of the diversity of synaptic timescales both in silico and in vivo (8, 29, 30). In addition, other works have validated and uncovered neural mechanisms observed in experimental settings using RNN models trained with backpropagation (7, 13, 37), thus highlighting that a network model can be biologically plausible even if it was constructed using nonbiological means. Another limitation of our method is the lack of temporal coding in our LIF models. Since our framework involves rate RNNs
that operate in a rate-coding scheme, the spiking RNNs that our framework produces also use rate coding by nature. Previous studies have shown that spike coding can improve spiking efficiency and enhance network stability (40, 38, 39), and recent studies emphasized the importance of precise spike coordination without modulations in firing rates (40, 41). Lastly, our framework does not model nonlinear dendritic processes, which have been shown to play a significant role in efficient input integration and flexible information processing (22, 42, 43). Incorporating nonlinear dendritic processes into our platform using the method proposed by Thalmeier et al. (22) will be an interesting next step to further investigate the role of dendritic computation in information processing.

In summary, we provide an easy-to-use platform that converts a continuous recurrent network model with basic biological constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model. The tight relationship between rate and LIF RNN models under certain parameter values suggests constraints to a spiking model.

Materials and Methods

The implementation of our framework and the codes to generate all of the figures in this work are available at https://github.com/krim035/SpikyRNN. The repository also contains implementation of other tasks, including autonomous oscillation and delayed match-to-sample tasks.

All of the trained models used in this study have been deposited into Open Science Framework (44).

Continuous Rate Network Structure. The continuous rate RNN model contains $N$ units recurrently connected to one another. The dynamics of the model is governed by

$$\tau \frac{dx}{dt} = -x + W^{rate} r + I_{ext},$$

where $W^{rate} \in \mathbb{R}^{1 \times N}$ corresponds to the synaptic decay time constants for the $N$ units in the network (Training Details discusses how these are initialized and optimized), $x \in \mathbb{R}^{1 \times N}$ is the synaptic current variable, $W^{rate} \in \mathbb{R}^{N \times N}$ is the synaptic connectivity matrix, and $I^{rate} \in \mathbb{R}^{1 \times N}$ is the output of the units. The output of each unit, which can be interpreted as the firing rate estimate, is obtained by applying a nonlinear transfer function to the synaptic current variable ($x$) elementwise:

$$r^{rate} = \phi(x).$$

We use a standard logistic sigmoid function for the transfer function to constrain the firing rates to be nonnegative:

$$\phi(x) = \frac{1}{1 + \exp(-x)}.$$

The connectivity weight matrix ($W^{rate}$) is initialized as a random, sparse matrix drawn from a normal distribution with 0 mean and an SD of 1.5/\sqrt{N}$, where $P_{0.20}$ is the initial connectivity probability.

The external currents ($I^{ext}$) include task-specific input stimulus signals (SI Appendix) along with a Gaussian white noise variable:

$$I_{ext} = W_{in} u + \mathcal{N}(0, 0.01),$$

where the time-varying stimulus signals ($u \in \mathbb{R}^{1 \times 1}$) are fed to the network via $W_{in} \in \mathbb{R}^{N \times N_{in}}$, a Gaussian random matrix with 0 mean and unit variance. $N_{in}$ corresponds to the number of input signals associated with a specific task, and $\mathcal{N}(0, 0.01) \in \mathbb{R}^{1 \times 1}$ represents a Gaussian random noise with 0 mean and variance of 0.01.

The output of the rate RNN at time $t$ is computed as a linear readout of the population activity:

$$o^{rate}(t) = W^{rate} r^{rate}(t),$$

where $W^{rate} \in \mathbb{R}^{1 \times N}$ refers to the readout weights. Eq. 5 is discretized using the first-order Euler approximation method:

$$x_{t+1} = (1 - \Delta t/\tau) x_{t} + \Delta t/\tau (W^{rate} r^{rate}_{t-1} + W^{rate} \Delta t + I^{rate}_{t-1}),$$

where $\Delta t = 5$ ms is the discretization time step size used throughout this study.

Spiking Network Structure. For our spiking RNN model, we considered a network of LIF units governed by

$$\tau_m \frac{dv}{dt} = -v + W^{spike} spk + I_{ext},$$

In the above equation, $\tau_m = 10$ ms is the membrane time constant shared by all of the LIF units, $v \in \mathbb{R}^{1 \times N}$ is the membrane variable, $W^{spike} \in \mathbb{R}^{N \times N}$ is the recurrent connectivity matrix, and $I^{spike} \in \mathbb{R}^{1 \times N}$ represents the spike trains filtered by a synaptic filter. Throughout the study, the double-exponential synaptic filter was used to filter the presynaptic spike trains:

$$\frac{ds_{spk}}{dt} = -s_{spk} + \frac{1}{\tau_{spk}} \sum_{t \lt t} \delta(t - \tau_{spk}),$$

where $\tau_{spk} = 2$ ms and $\tau_{spk}^r$ refer to the synaptic rise time and the synaptic decay time for unit $i$, respectively. The synaptic decay time constant values ($\tau_{spk}^r$) are $t_{spk}^r$ defined as

$$s_{spk} = \tau_{spk} \sum_{t \lt t} \delta(t - \tau_{spk}),$$

where $\tau_{spk} = 2$ ms and $\tau_{spk}^r$ refer to the synaptic rise time and the synaptic decay time for unit $i$, respectively. The synaptic decay time constant values ($\tau_{spk}^r$) are $t_{spk}^r$ defined as

$$s_{spk} = \tau_{spk} \sum_{t \lt t} \delta(t - \tau_{spk}),$$

where $\tau_{spk} = 2$ ms and $\tau_{spk}^r$ refer to the synaptic rise time and the synaptic decay time for unit $i$, respectively. The synaptic decay time constant values ($\tau_{spk}^r$) are $t_{spk}^r$ defined as

$$s_{spk} = \tau_{spk} \sum_{t \lt t} \delta(t - \tau_{spk}),$$

where $\tau_{spk} = 2$ ms and $\tau_{spk}^r$ refer to the synaptic rise time and the synaptic decay time for unit $i$, respectively. The synaptic decay time constant values ($\tau_{spk}^r$) are $t_{spk}^r$ defined as

$$s_{spk} = \tau_{spk} \sum_{t \lt t} \delta(t - \tau_{spk}),$$

where $\tau_{spk} = 2$ ms and $\tau_{spk}^r$ refer to the synaptic rise time and the synaptic decay time for unit $i$, respectively. The synaptic decay time constant values ($\tau_{spk}^r$) are $t_{spk}^r$ defined as

$$s_{spk} = \tau_{spk} \sum_{t \lt t} \delta(t - \tau_{spk}),$$

where $\tau_{spk} = 2$ ms and $\tau_{spk}^r$ refer to the synaptic rise time and the synaptic decay time for unit $i$, respectively. The synaptic decay time constant values ($\tau_{spk}^r$) are $t_{spk}^r$ defined as
where \( \sigma() \) is the sigmoid function (identical to Eq. 6) used to constrain the time constants to be nonnegative. The time constant values are also bounded by the minimum \( \tau_{\text{down}} \) and the maximum \( \tau_{\text{up}} = \tau_{\text{down}} + \tau_{\text{max}} \) values. The error computed from the loss function (Eq. 8) is then backpropagated to update the time constants at each iteration:

\[
\frac{\partial L}{\partial \tau_j} = \frac{\partial L}{\partial r_j} \frac{\partial r_j}{\partial \tau_j} + \frac{\partial L}{\partial \sigma(r_j)} \frac{\partial \sigma(r_j)}{\partial \tau_j}.
\]

The method proposed by Song et al. (13) was used to impose Dale’s principle and create separate excitatory and inhibitory populations. Briefly, the recurrent connectivity matrix (\( W^{\text{rate}} \)) in the rate model is parameterized by

\[
W^{\text{rate}} = \left[ W^{\text{rate}} \right]_+ \cdot D,
\]

where the rectified linear operation \([\cdot]_+ \) is applied to the connectivity matrix at each update step. The diagonal matrix \( D \in \mathbb{R}^{N \times N} \) contains \( +1 \) for excitatory units and \(-1 \) for inhibitory units in the network. Each unit in the network is randomly assigned to 1 group (excitatory or inhibitory) before training, and the assignment does not change during training (i.e., \( D \) stays fixed).

To impose specific connectivity patterns, we apply a binary mask (\( M \in \mathbb{R}^{N \times N} \)) to Eq. 9:

\[
W^{\text{spk}} = \left[ W^{\text{rate}} \right]_+ \cdot D \odot M,
\]

where \( \odot \) refers to the Hadamard operation (elementwise multiplication). Similarly to the diagonal matrix \( D \), the mask matrix stays fixed throughout training. For example, the following mask matrix can be used to create a subgroup of inhibitory units (Group A) that do not receive synaptic inputs:

\[
M_{ij} = \begin{cases} 0 & i \in \text{Group A}, j \in \text{Group B} \\ 1 & \text{otherwise} \end{cases}
\]

where \( M_{ij} \in M \) establishes (if \( M_{ij} = 1 \)) or removes (if \( M_{ij} = 0 \)) the connection from unit \( j \) to unit \( i \).

Transfer Learning from a Rate Model to a Spiking Model. In this section, we describe the method that we developed to perform transfer learning from a trained rate model to a LIF model. After the rate RNN model is trained using the gradient descent method, the rate model parameters are transferred to an LIF network in a one-to-one manner. First, the LIF network is initialized to have the same topology as the trained rate RNN. Second, the input weight matrix (\( W_i \)) and the synaptic decay time constants (\( \tau_i \)) are transferred to the spiking RNN without any modification. Third, the recurrent connectivity matrix (\( W^{\text{rate}} \)) and the readout weights (\( W^{\text{out}} \)) are scaled by a constant number, \( \lambda \), and transferred to the spiking network.

If the recurrent connectivity weights from the trained rate model are transferred to a spiking network without any changes, the spiking model produces largely fluctuating signals (as illustrated in Fig. 2B), because the LIF firing rates are significantly larger than 1 (whereas the firing rates of the rate model are constrained to range between 0 and 1 by the sigmoid transfer function).

To place the spiking RNN in the similar dynamic regime as the rate network, we first assume a linear relationship between the rate model connectivity weights and the spike model weights:

\[
W^{\text{spk}} = \lambda \cdot W^{\text{rate}}.
\]

Using the above assumption, the synaptic drive (\( \mathbf{d} \)) that unit \( i \) in the LIF RNN receives can be expressed as

\[
\mathbf{d}_i(t) = \sum_{j=1}^{N} W^{\text{spk}}_{ij} \cdot r_j(t) = \sum_{j=1}^{N} \left( \lambda \cdot W^{\text{rate}}_{ij} \right) \cdot r_j(t).
\]

where \( W^{\text{spk}} \in \mathbb{R}^{N \times N} \) is the synaptic weight from unit \( j \) to unit \( i \). Similarly, unit \( i \) in the rate RNN model receives the following synaptic drive at time \( t \):

\[
\mathbf{d}_i^{\text{rate}}(t) = \sum_{j=1}^{N} W^{\text{rate}}_{ij} \cdot r_j^{\text{rate}}(t).
\]

If we set the above 2 synaptic drives (Eqs. 10 and 11) equal to each other, we have

\[
\mathbf{d}_i^{\text{spk}}(t) = \mathbf{d}_i^{\text{rate}}(t)
\]

Generalizing Eq. 12 to all of the units in the network, we have

\[
\mathbf{r}^{\text{rate}}(t) = \lambda \cdot \mathbf{r}^{\text{spk}}(t).
\]

Therefore, if there exists a constant factor \( \lambda \) that can account for the firing rate scale difference between the rate and the spiking models, the connectivity weights from the rate model (\( W^{\text{rate}} \)) can be scaled by the factor and transferred to the spiking model.

The readout weights from the rate model (\( W^{\text{out}} \)) are also scaled by the same constant factor \( \lambda \) to have the spiking network produce output signals similar to the ones from the trained rate model:

\[
\mathbf{o}^{\text{rate}}(t) = \lambda \cdot \mathbf{o}^{\text{spk}}(t)
\]

In order to find the optimal scaling factor, we developed a simple grid search algorithm. For a given range of values for \( 1/\lambda \) (ranged from 20 to 75 with a step size of 5), the algorithm finds the optimal value that maximizes the task performance.

ACKNOWLEDGMENTS. We thank Ben Huh, Gerald Pao, Jason Fleischer, Debra Amaya, Yusi Chen, and Ben Tsuda for helpful discussions and feedback on the manuscript. We also thank Jorge Aldana for assistance with computing resources. This work was funded by National Institute of Mental Health Grant F30MH115605-01A1 (to R.K.), the Harold R. Schwanenberg Medical Scholarship (R.K.), and the Bunnard–Partridge Foundation Scholarship (R.K.). We acknowledge the support of the NVIDIA Corporation with the donation of the Quadro P6000 graphics processing unit used for this research. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.