Recurrent networks recognize patterns with low-dimensional oscillations

1st Keith T. Murray
Department of Brain and Cognitive Sciences
Massachusetts Institute of Technology
Cambridge, USA
ktmurray@mit.edu

***Abstract*—This study proposes a novel mechanism for pattern recognition using a Recurrent Neural Network (RNN) trained on a task inspired by the SET card game. The network recognizes patterns via phase shifts in a low-dimensional limit cycle, modeled as transitions within a Finite State Automaton (FSA). This work not only provides a potential neural interpretation of FSAs but also deepens our understanding of the cognitive computations underpinning pattern recognition. Furthermore, the interpretability of mechanisms learned by RNNs is emphasized, contributing to the discourse on deep learning model interpretability.**

***Index Terms*—Pattern recognition, Recurrent neural networks, Neural dynamics, Cognitive modeling, Deep learning interpretability**

I. Introduction

Pattern recognition underpins human cognition, influencing perception, language, and reasoning. Despite significant investigations into humans' pattern recognition abilities [1], the computational underpinnings within neuronal circuits remain elusive. Recent developments in recurrent neural networks (RNNs) have generated theories on neuronal mechanisms underlying various cognitive abilities [2]–[5]. This study capitalizes on these developments, proposing an innovative pattern recognition task solved by an RNN through phase shifts in a low-dimensional oscillator. This mechanism is interpreted as transitions in a finite state automata (FSA), presenting a potential neural implementation of FSAs. We also present a simple handcrafted oscillatory model capturing the low-dimensional dynamics of the fully trained model. This research pushes the boundary of our understanding in neural computations for pattern recognition, offering new insights into deep learning model interpretability.

II. Background

A. Dynamical motifs in RNNs

Previous work by [4] showed that RNNs trained on many cognitive tasks learned a series of modular, low-dimensional, dynamical phenomena (e.g. attractors, limit cycles, bifurcations [6]), termed *dynamical motifs*, that were shared among all cognitive tasks. In addition, simple tasks would learn simple dynamical motifs and complex tasks would reuse multiple dynamical motifs form simple tasks and combine them in a manner apt for performing in the complex task. This work suggests that all cognitive abilities have associated simple dynamical phenomena that are computed at the neuronal level. It is possible that the brain only computes a handful of simple dynamical phenomena that can be combined and reused to perform all cognitive tasks.

B. Dynamical motifs for transitive inference

Previous work by [5] showed that RNNs trained to perform a transitive inference task learned a simple dynamical mechanism characterized by a single oscillation and a collinear encoding of stimulus inputs. The oscillatory activity of the dynamical mechanism would subtract the encoded magnitudes of stimulus inputs in order to perform transitive inference. Ref. [5] hypothesized this dynamical mechanism to underlie transitive inference in cognitive agents. Other higher-order cognitive abilities incorporating transitive inference might also incorporate this particular mechanism. Building on this interpretation and motivation, we sought to uncover the dynamical mechanism underlying pattern recognition through training RNNs and reverse-engineering their learned representations.

C. Why do interpretable dynamical motifs emerge?

The simplicity of the dynamical mechanisms learned in [4] and [5] is surprising. The tricks required for RNNs to learn these motifs are large regularizations imposed during training. Both works included an L2 penalty on the trained parameters and a metabolic penalty on the recurrent activations. In this work, we used these regularizations in order to bias our model to learn simple dynamical representations.

III. Methodology

Our methodology is segmented into three parts: defining our novel pattern recognition task (sec. III-A), defining the model and its parameters (sec. III-B), and describing our analysis methods (sec. III-C).

A. Task

While there exists a host of pattern recognition and classification tasks in the machine learning community, their associated datasets may be too complex or noisy to elicit interpretable dynamical motifs. We instead took inspiration for the design of our own task from the card game SET [7]. SET is a card game where players try to identify sets of three cards out of a field of 12 cards. Each card has four attributes: shape, color, number, and shading. Each of these attributes has

3 possible values. A set is found when each attribute across all cards is entirely *uniform* or *heterogeneous*.

Our task was concerned with validating a set with only one attribute. We refer to this singular attribute as *color*, and the three values associated are *green*, *purple*, and *red*. The task involved sequentially presenting colors to an agent during a trial and having the agent classify at the end of a trial if the colors presented constituted a valid SET (i.e. the agent decides if all the colors presented are the same or different).

All trials of the task were $500ms$. There was a delay period of $30ms$ at the beginning and $100ms$ at the end of the trial where no colors were presented. A color was presented for $20ms$, and there was at least a $30ms$ gap between the presentation of colors (Fig. 1). Given that the agent was an RNN, each color was encoded as a 100-dimensional vector[1] with values drawn from a random distribution[2], and the RNN produced an output value of $+1$ to indicate a valid pattern and -1 to indicate an invalid pattern.

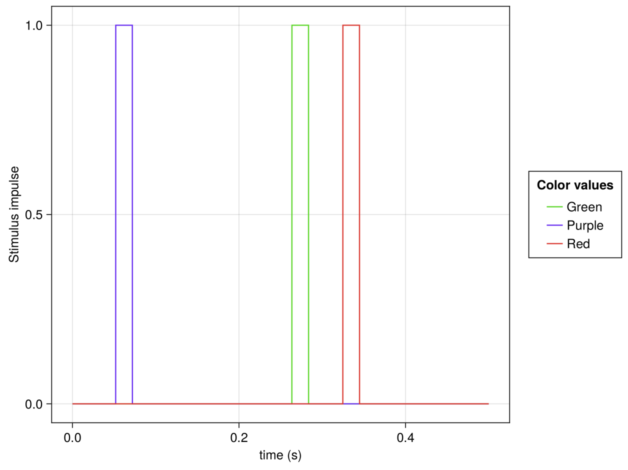

Fig. 1. A sample trial showing the colors purple, green, and red being presented. This trial constitutes a valid pattern.

The model was trained on a generated set of 540 trials. The proportion of accepting to rejecting stimuli in the training SET was 50%. 27 testing trails were generated where each testing trial was a different pattern of stimuli. Trials were mini-batched during training into batches of 108 trials.

B. Model

We describe our model according to the framework proposed in [8].

Architecture: The model was a standard continuous-time RNN:

$$\tau \dot{x}_i(t) = -x_i(t) + \sum_{k=1}^{N} J_{ik} r_k(t) + \sum_{k=1}^{N^{in}} B_{ik} u_k(t) + b_i + \eta_i(t) \quad (1)$$

$$r_i(t) = \tanh(x_i) \quad (2)$$

$$z(t) = \sum_{k=1}^{N} W_k r_k(t) + b_{out} \quad (3)$$

τ is interpreted as the time constant for the RNN and was set to $10ms$. $x_i(t)$ is interpreted as the average voltage value for the ith subpopulation of cortical neurons at time t. N is the total number of subpopulations of cortical neurons and was set at 100. $u_k(t)$ is interpreted as the input stimulus function conveying color values. N^{in} is the number of input dimensions and was, as stated in section 3.1, set at 100. $\eta_i(t)$ is a random value drawn from a Gaussian distribution, $\mathcal{N}(0, 0.10)$. $r_i(t)$ is interpreted as the average firing rate value for the ith subpopulation of cortical neurons. The activation function that converts voltage to firing rate is the hyperbolic tangent (tanh). $z(t)$ is the value of the output neuron.

Equation (1) was approximated with Euler's method and a step size (Δt) of $10ms$ for all trials. The entirety of our model was coded in Julia [9] using the Lux framework [10] and SciML ecosystem [11], [12].

Learning algorithm: We used the AdamW learning algorithm [13] and reverse mode automatic differentiation to train the adjustable parameters that define our neural network: the initial state $\mathbf{x}(t=0)$, recurrent matrix $\mathbf{J}$, input matrix $\mathbf{B}$, input bias vector $\mathbf{b}$, internal weight matrix $\mathbf{W}$, and output bias b_{out}.. The initial learning rate was set to 10^{-4}. L2 regularization, as mentioned in sec. II-C, was implemented as a weight decay regularization in the AdamW algorithm and was set to 10^{-4}.

Objective function: We used a mean squared error (MSE) objective function to compare the observed $z(t)$ value to the expected $\hat{z}(t)$ value. The last 5 time steps of the $500ms$ trail were used for the MSE function. Metabolic regularization, as mentioned in sec. II-C, was added to the objective function to bias the recurrent activations toward being low in amplitude. The full objection function was the following:

$$\mathcal{L}(\hat{z}, z, \mathbf{r}) = \frac{1}{5} \sum_{t=T-5}^{T} (\hat{z}(t) - z(t))^2 + \lambda \frac{1}{TN} \sum_{t,i=1}^{T,N} r_i^2(t) \Delta t \quad (4)$$

T is the total number of time steps, 50 for all trials. λ is the metabolic penalty and was set to 10^{-4}.

C. Analysis

To analyze the learned representations of our model, we primarily utilized Principal Component Analysis (PCA), a popular dimensionality reduction technique. PCA has been prevalently employed in previous RNN research for visualizing the low-dimensional dynamics of RNNs [2]–[5]. We performed PCA on the firing rates of our model, $\mathbf{r}(t)$, under the condition of no recurrent noise, i.e., $\eta_i(t) = 0$.

It is essential to mention that PCA mainly discerns the correlations amongst the firing rates of our model, rather than identifying causal, mechanistic properties [14]. To address this inherent limitation of PCA, we developed a simplified model of the low-dimensional rates within our network. This

[1]Our use of a 100-dimensional vector was inspired by [5]. This dimensionality ensured that stimulus representations were uncorrelated in the activity space of the model.

[2]The same encoding vectors were used across all training and testing trials.

approach allows us to confirm the existence of the mechanisms suggested by the PCA analysis.

Our initial hypothesis postulated that PCA would reveal an underlying fixed-point network in the shape of a hexagon (Fig. 2). Drawing inspiration from the Hopfield network [15] and insights provided in [16], we conceived this notion of a hexagonal fixed-point network.

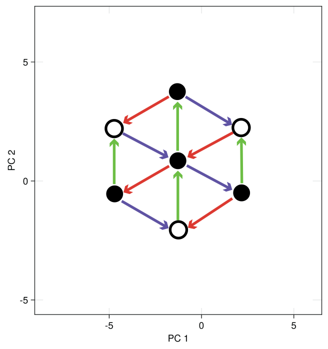

Fig. 2. An illustration of our hypothesized low-dimensional, learned representation in the form of a hexagonal fixed-point network. Filled fixed points represent accepted patterns, while empty ones denote rejected patterns. Note that all transitions triggered by a specific color stimulus maintain a consistent directionality. The architecture of this network allows for linear segregation of accepting and rejecting fixed points in PC 3, simplifying classification tasks.

In this hypothesized network, activity originates in the center with the sequential introduction of a color stimulus sequentially propelling the network's activity from one fixed-point to another. Note the consistent directionality associated with each color in the graph. Color transitions absent from the diagram are represented as self-loops. In PC 3, the fixed-points corresponding to acceptance and rejection can be linearly segregated, simplifying the classification process.

This hexagonal fixed-point network can be represented as an FSA, with valid patterns represented as a subset of the regular language of the FSA [17].

IV. Results

A. Accuracy and Preliminary Analysis

The fully-trained RNN model demonstrated robust performance, achieving an accuracy of 96.30% in training and 100.00% in testing. In the absence of recurrent noise, the model displayed flawless performance, reaching an accuracy of 100.00% in both training and testing. Despite this perfect accuracy, it was somewhat anticipated due to the straightforward nature of the dataset. The model's output exhibited an oscillatory pattern, with *perturbations* corresponding to the introduction of the stimulus (Fig. 3).

For valid patterns, the model's output completes the trial in the positive phase of its underlying oscillation, generating a value of $+1$. Conversely, for invalid patterns, the output concludes the trial in the negative phase of its oscillation, yielding a value of -1. In the absence of a stimulus input, the model is observed to complete two oscillations within a single trial. The *perturbations* corresponding to stimulus presentation

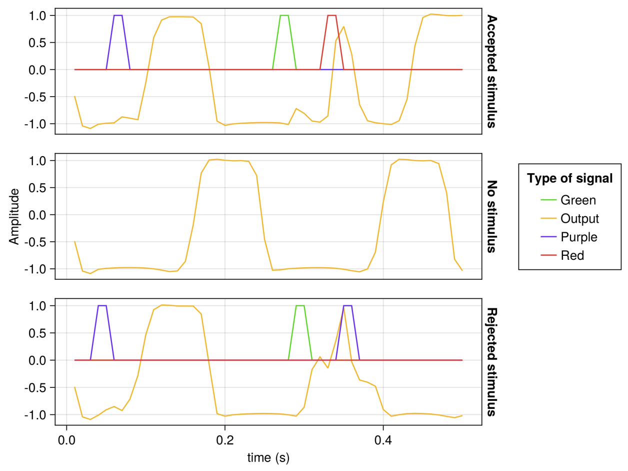

Fig. 3. Three examples of stimuli presented during trials and the models associated output. The top row shows a valid pattern trial, resulting in a model output of $+1$. The middle row presents a trial with no color stimulus and demonstrates the model completing two full oscillations in a single trial. The bottom row shows an invalid pattern trial, resulting in a model output of -1. Note the *perturbations* due to stimulus presentation in the top and bottom rows.

indicate the influence of the stimulus on the oscillation, yet their computational significance remains unclear without further PCA analysis.

B. PCA results

The PCA of the model's firing rates revealed that the model had learned a low-dimensional limit cycle[3] that completed two full rotations during a trial (Fig. 4). At the end of the trial, PCA reveals that the activations of the model would lie in one of three distinct locations along the limit cycle. Two locations represent invalid patterns while one location represents valid patterns. This dynamical behavior indicates that stimulus would shift the phase of the limit cycle in the model. This may be the computational affect of the *perturbations* seen in Fig. 3.

To confirm that stimulus presentation resulted in phase-shifts along the limit cycle in the model, we further investigated the low-dimensional trajectories of the model by rotating the PCA data and visualizing the rotated PCA trajectories. Notice how in Fig. 4 if data were to be projected onto principal component (PC) 1, then the distribution of valid pattern PCA endpoints would mix with the distribution of invalid pattern PCA endpoints. By rotating PCA data by 60 degrees, the distributions of valid and invalid pattern endpoints are linearly separable when projected onto PC 1.

Fig. 5 presents the dynamics of the model projected onto PC 1 over time for various trials. By comparing the impact of stimuli from different trials on the dynamics, we can formulate preliminary theories about how stimulus presentation influences the dynamics of the underlying limit cycle.

A trial with all green stimuli appears to only slightly perturb the model's dynamics compared to a trial with no stimulus

[3]Refer to [6] for an overview of limit cycles in nonlinear dynamics.

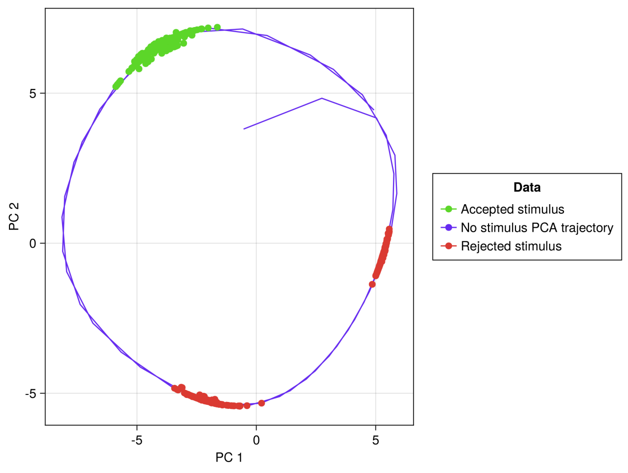

Fig. 4. This figure illustrates a low-dimensional representation of the model's trajectories, showing distinct clusterings at the endpoints of both accepted and rejected patterns. The trajectory validates the underlying dynamics of the model as a limit cycle. The clusterings at the endpoints of trajectories for accepted and rejected patterns imply that stimulus presentations induce phase shifts along this limit cycle.

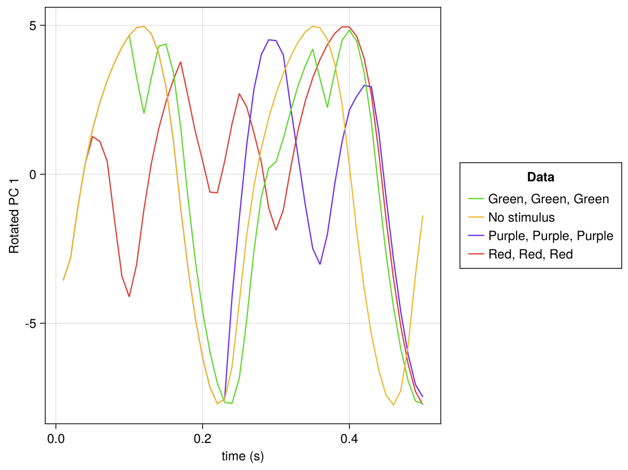

Fig. 5. This figure depicts the model's firing rates across four trials with various stimuli, projected onto the rotated PC 1 over time. The influence of stimulus presentation on the dynamics of the rotated PC 1 is visible. The Green, Green, Green trial shows minimal change compared to the no stimulus trial. When comparing the Red, Red, Red trial to the Purple, Purple, Purple trial, it appears that the red stimulus may reduce the phase of the oscillation while the purple stimulus may increase it.

(Fig. 5). Red stimuli seem to briefly *reset* the network's dynamics, whereas purple stimuli cause the oscillation to *jump* in phase. While these initial theories are somewhat broad, they offer a starting point for further investigation.

In light of the initial hypothesis positing a FSA with nodes representing fixed-points in the model, the theories developed from Fig. 5 prompt us to update our hypothesis. Given the presence of three distinct endpoint distributions (Fig. 4) and phase-shifts marking transitions between these distributions, our updated FSA now includes three nodes. In this updated model, stimulus input induces transitions between nodes, and each node corresponds to a specific phase angle, instead of a fixed-point, within the model (Fig. 6). We have revised our initial assumption that all stimulus inputs of a particular color should point in the same direction. Instead, we now propose that all stimulus inputs must similarly affect the phase of the oscillation.

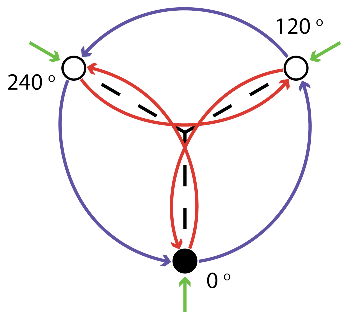

Fig. 6. This figure depicts the updated FSA that represents the underlying dynamics and computation of our trained model. The model begins at a phase angle of 0 degrees, and stimulus presentations trigger transitions to other nodes, each representing a distinct phase angle.

While we previously observed that green stimuli do impact the dynamics of the model (Fig. 3), we simplify our model by assuming that green stimuli do not affect the underlying phase angle. Purple stimuli are postulated to add 120 degrees to the phase angle, while red stimuli subtract 120 degrees. If all stimuli are either *uniform* or *heterogeneous*, the phase angle shifts sum to 0, causing the oscillation to land in an accepting phase.

To test the accuracy of this FSA model, we made predictions about the phase at which the model will conclude for invalid stimuli. For instance, if the stimulus pattern is purple, green, purple, then the model should end 120 degrees behind the accepting patterns. If the stimuli is red, green, red, then the model should end 120 degrees ahead of the accepting patterns. Fig. 7 confirms these predictions and supports the validity of our new FSA model.

C. *Simplifying the model*

To deepen our understanding of the underlying computations in our model, we developed a new, simpler model that could replicate the dynamics of the original one. This process would further validate the FSA representation of our model's pattern recognition capabilities (Fig. 6). The dynamics of the rotated PC 1 of our model can be expressed with the following equation:

$$f(t) = 7\sin(\frac{2\pi}{0.29}t + \sum_{t_o=1}^{t} \phi(t_o)) - 1.5 \tag{5}$$

In (5), $\phi(t)$ represents the pattern signal, which is $\frac{2\pi}{3}$ if the stimulus is purple, $-\frac{2\pi}{3}$ if the stimulus is red, and 0 if the stimulus is green or if no stimulus is present.

It is important to note that (5) simplifies many of the dynamical properties observed in the dynamics of rotated PC 1. We simplify the frequency of (5) to be constant while it is

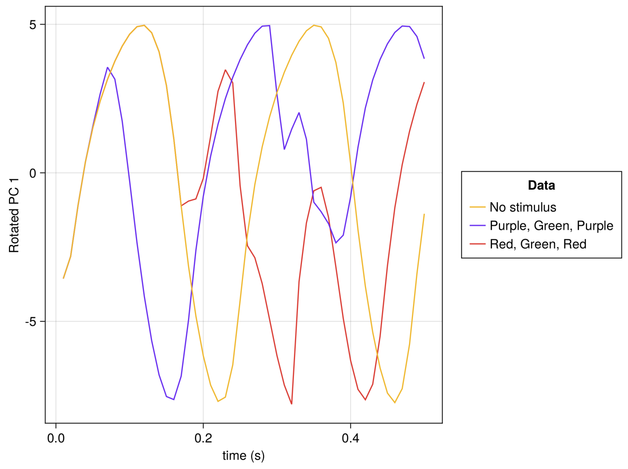

Fig. 7. This figure illustrates the projected trajectories of several invalid patterns onto the rotated PC 1. The depicted trajectories confirm the model's behavior according to our updated FSA, validating our predictions about the final phase for different stimulus sequences.

evident that the rate of the dynamics around the limit cycle is nonuniform (Fig. 4). In addition, the presentation of stimulus causes a dynamical response in the dynamics of the full model (Fig. 5) and (5) models these responses as being sharp transitions in phase. Yet, with these simplifications, we believe that (5) captures the underlying computational mechanisms identified in the rotated PC 1.

As expected, this simplified model can accurately classify all trials, assuming there is no recurrent noise. Fig. 8 displays a few examples of the dynamics of (5) plotted alongside the actual dynamics of the Rotated PC 1 of the full model. The similarity between the trajectories of these two models provides additional validation for our FSA representation of the computations performed by our full model.

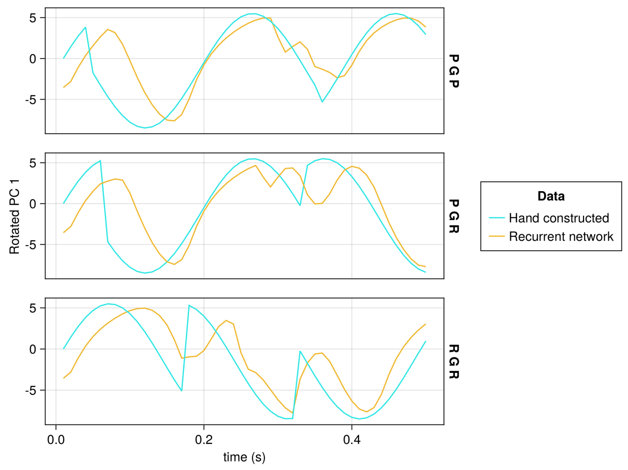

Fig. 8. This figure illustrates three examples of the trajectories of our simplified model in comparison to the rotated PC 1 of the full model. The top row displays both models in a purple, green, purple trial. The middle row shows both models in a purple, green, red trial. The bottom row represents both models in a red, green, red trial.

V. Discussion

In this study, we reveal a key mechanism through which an RNN, trained for pattern recognition in a task inspired by the SET card game, accurately recognizes all patterns devoid of noise. By interpreting pattern recognition through phase angle shifts in a low-dimensional limit cycle as transitions within a finite state automaton (FSA), we present a fresh perspective on the neural computations that underpin pattern recognition (Fig. 6). Notably, this finding underscores the ability to interpret the mechanisms learned by an RNN, a theme we will elaborate on throughout this discussion.

While there is a lengthy precedent of FSAs being employed to model human cognition [18], our knowledge has yet to widely extend to the neural circuit-level implementation of these FSAs. Our study fills this gap, presenting an FSA learned within an RNN, a biologically plausible model. Nevertheless, we caution that this finding primarily pertains to a single attribute and simple *uniform* or *heterogeneous* patterns, necessitating further investigation to ascertain its applicability to more complex pattern recognition problems.

This work's significance lies in contributing to an expanding understanding of the neural computations underpinning pattern recognition, paving the way toward a more comprehensive grasp of neural computation. Echoing the findings from [3], the dynamical mechanisms unveiled in this study could likely be found in the prefrontal cortex of primates executing similar pattern recognition tasks. While prior studies provided evidence of accumulation and decision-making [16], the complexity of stimuli was relatively simple. Only biological evidence can determine whether the neural dynamics underlying pattern recognition are oscillatory or attractive.

We ambitiously look forward to extending this work by exploring the integration of oscillators with other dynamical motifs to model higher-order cognitive behaviors [4]. This parallels the compilation of logical operations into algorithms and computer programs, underscoring the analogies between biological and artificial computational systems [19].

Furthermore, in the context of neural circuit-level FSA implementations, we note our work's novelty. Previously, it was postulated that a Hopfield network could be represented as the average phase in a system of weakly coupled oscillators [20]. Existing work also demonstrated RNNs' ability to retain stimuli in the phase shift of a coupled oscillator system [21]. We build on these studies to show that RNNs can implement sequential pattern recognition in limit cycles like FSAs, potentially hinting at computational mechanisms for brain oscillations.

Contrasting with other computational neuroscience work proposing oscillatory computational mechanisms, which often presumes neurons to oscillate [20]. or receive oscillatory input [21], our model makes no such assumptions and learns an oscillatory mechanism. However, our work's limitations should be acknowledged: PCA only uncovers correlations without revealing causal mechanisms [14], and the recognized patterns are rule-based and simple.

The emergence of limit cycles rather than attractor networks, owing to the compact realization of FSAs, raises a fascinating question. The observed oscillatory activity may be attributed to task-specific implementations. Given that RNNs are universal function approximators [22], there could be some parameters of our task that realize our initial hypothesis in a trained RNN (Fig. 2).

Future investigations could focus on developing a more descriptive simplified model of the RNN that incorporates the dynamics of phase shifts induced by stimuli, instead of instantaneous shifts (Fig. 8). They could also explore how these FSA models of pattern validation integrate into visual search for comprehensive SET playing, and the potential application of cognitive technologies using oscillations for cognitive-like computations.

Our work not only offers new insights into the neural computations underpinning pattern recognition but also illuminates the interpretability of mechanisms learned by RNNs. In a field where interpretability is becoming increasingly valuable, this serves as a clear and important example of how these mechanisms can be easily understood and interpreted. This will undoubtedly be of interest to the machine learning community, which is making concerted efforts to understand the mechanisms in deep learning models [23]. In sum, this study presents a significant stride toward a more comprehensive understanding of neural computations and their implementation in pattern recognition tasks.

Acknowledgment

We thank Nikasha Patel, Brabeeba Wang, Sabrina Drammis, and Nancy Lynch for intellectual and helpful discussions pertaining to the conceptualization of this work. We would also like to acknowledge the assistance of OpenAI's ChatGPT in drafting and editing portions of the manuscript.

References

[1] B. Inhelder and J. Piaget, "The Early Growth of Logic in the Child: Classification and Seriation," London, England: Psychology Press, 1964.

[2] D. Sussillo and O. Barak, "Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks," Neural computation, vol. 25, no. 3, pp. 626–649, 2013.

[3] V. Mante, D. Sussillo, K. V. Shenoy, and W. T. Newsome, "Context-dependent computation by recurrent dynamics in prefrontal cortex," Nature, vol. 503, no. 7474, pp. 78–84, 2013.

[4] L. Driscoll, K. Shenoy, and D. Sussillo, "Flexible multitask computation in recurrent networks utilizes shared dynamical motifs," bioRxiv, 2022.

[5] K. Kay, X. Wei, R. Khajeh, M. Beiran, C. J. Cueva, G. Jensen, V. P. Ferrera, and L. F. Abbott, "Neural dynamics and geometry for transitive inference," bioRxiv, 2022.

[6] S. H. Strogatz, "Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, And Engineering," CRC press, 2000.

[7] L. McMahon, G. Gordon, H. Gordon, and R. Gordon, "The Joy of SET: The Many Mathematical Dimensions of a Seemingly Simple Card Game," Princeton University Press, 2017.

[8] B. A. Richards, T. P. Lillicrap, P. Beaudoin, Y. Bengio, R. Bogacz, A. Christensen, C. Clopath, R. P. Costa, A. de Berker, S. Ganguli, et al., "A deep learning framework for neuroscience," Nature neuroscience, vol. 22, no. 11, pp. 1761–1770, 2019.

[9] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," SIAM review, vol. 59, no. 1, pp. 65–98, 2017.

[10] A. Pal, "Lux: Explicit Parameterization of Deep Neural Networks in Julia," GitHub repository, 2022, [Online]. Available: https://github.com/avik-pal/Lux.jl/.

[11] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, and A. Edelman, "Universal differential equations for scientific machine learning," arXiv preprint arXiv:2001.04385, 2020.

[12] C. Rackauckas and Q. Nie, "Differentialequations. jl–a performant and feature-rich ecosystem for solving differential equations in julia," Journal of open research software, vol. 5, no. 1, 2017.

[13] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," arXiv preprint arXiv:1711.05101, 2017.

[14] C. Langdon and T. A. Engel, "Latent circuit inference from heterogeneous neural responses during cognitive tasks," bioRxiv, 2022.

[15] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," Proc. Natl. Acad. Sci., vol. 79, no. 8, pp. 2554–2558, 1982.

[16] W. Chaisangmongkon, S. K. Swaminathan, D. J. Freedman, and X.-J. Wang, "Computing by robust transience: how the fronto-parietal network performs sequential, category-based decisions," Neuron, vol. 93, no. 6, pp. 1504–1517, 2017.

[17] P. Tiňo, B. G. Horne, C. L. Giles, and P. C. Collingwood, "Finite state machines and recurrent neural networks—automata and dynamical systems approaches," in Neural networks and pattern recognition, Elsevier, 1998, pp. 171–219.

[18] E. L. Antworth, "PC-KIMMO: a two-level processor for morphological analysis," Summer Institute of Linguistics, 1990.

[19] H. Jaeger, "Towards a generalized theory comprising digital, neuromorphic and unconventional computing," Neuromorphic Computing and Engineering, vol. 1, no. 1, pp. 012002, 2021.

[20] F. C. Hoppensteadt and E. M. Izhikevich, "Oscillatory neurocomputers with dynamic connectivity," Physical Review Letters, vol. 82, no. 14, pp. 2983, 1999.

[21] M. Pals, J. H. Macke, and O. Barak, "Trained recurrent neural networks develop phase-locked limit cycles in a working memory task," bioRxiv, 2023.

[22] K. Funahashi and Y. Nakamura, "Approximation of dynamical systems by continuous time recurrent neural networks," Neural Networks, vol. 6, no. 6, pp. 801–806, 1993.

[23] N. Nanda, L. Chan, T. Liberum, J. Smith, and J. Steinhardt, "Progress measures for grokking via mechanistic interpretability," arXiv preprint arXiv:2301.05217, 2023.