Inferring dynamical features from neural data through joint learning of latent factors and weights

Anirudh Jamkhandi ANIRUDH.JAMKHANDI@UMONTREAL.CA
Ali Korojy ALI.KOROJY@UMONTREAL.CA
Olivier Codol OLIVIER.CODOL@GMAIL.COM
Guillaume Lajoie* GUILLAUME.LAJOIE@MILA.QUEBEC
Matthew Perich* MATTHEW.PERICH@UMONTREAL.CA
*** co-senior authors**

Editors: List of editors' names

Abstract

Behavior arises from coordinated synaptic changes in recurrent neural populations. Inferring the underlying dynamics from limited, noisy, and high-dimensional neural recordings is a major challenge, as experimental data often provides only partial access to brain states. While data-driven recurrent neural networks (dRNNs) have been effective for modeling such dynamics, they are typically limited to single-task domains and struggle to generalize across behavioral conditions. Here, we propose a hierarchical model that captures neural dynamics across multiple behavioral contexts by learning a shared embedding space over RNN weights. We demonstrate that our model captures diverse neural dynamics with a single, unified model using both simulated datasets of many tasks and neural recordings during monkey reaching. Using the learned task embeddings, we demonstrate accurate classification of dynamical regimes and generalization to unseen samples. Crucially, spectral analysis on the learned weights provide valuable insights into network computations, highlighting the potential of joint embedding–weight learning for scalable inference of brain dynamics.

Keywords: Recurrent Neural Networks, Edge of Stability, Spectral Analysis, Neural Dynamics

1. Introduction

The brain is a complex nonlinear dynamical system, whose output is dependent on time-varying interactions among large neural populations (Duncker and Sahani, 2021). Advances in neural recording technologies provide access to increasingly large population activity, offering a window into neural computation across tasks and brain regions. A central challenge in modern computational neuroscience is how to leverage these large-scale recordings to reveal the dynamics underlying behavior.

Recurrent neural networks (RNNs) have emerged as a leading model for studying neural population activity (Sussillo and Abbott, 2009; Pandarinath et al., 2018; Hess et al., 2023; Dinc et al., 2023; Valente et al., 2022; Perich et al., 2021; Durstewitz et al., 2023). In particular, data-constrained RNNs explicitly fit their dynamics to large-scale neural recordings, enabling them to capture the latent computational structure underlying observed activity. However, these models are typically trained with a single, fixed set of weights and therefore yield only one dynamical mechanism, even though many distinct solutions could support

the same behavior (Turner et al., 2021). In contrast, biological circuits can flexibly reorganize their dynamics across contexts and tasks, engaging multiple dynamical regimes to support diverse behaviors. Standard RNNs, by comparison, often fail to generalize beyond the specific contexts or task conditions included in training. This gap highlights the need for modeling frameworks that capture context-dependent variation and represent families of task-relevant dynamical solutions.

Thus, we propose a hierarchical hypernetwork framework that simultaneously reconstructs neural activity and infers latent contexts. By generating the parameters of data-constrained RNNs, the hypernetwork(HNET) links RNN weights to their contextual representations. The shared hypernetwork captures common computational motifs, while context embeddings flexibly modulate task-specific dynamics.

2. Methods

Here, we describe our approach to learning a unified dynamical model from neural trajectories by fitting RNNs to recorded data conditioned on contexts. We use a rate-based RNN with N units, where activity $\mathbf{r}(t) \in \mathbb{R}^N$ evolves as

$$\tau \frac{d\mathbf{r}}{dt} = -\mathbf{r}(t) + \mathbf{J}\phi(\mathbf{r}(t)) + \xi(t), \tag{1}$$

with time constant $\tau > 0$, $\mathbf{J} \in \mathbb{R}^{N \times N}$ is recurrent connectivity matrix, $\phi(x) = \tanh(x)$ is non-linear activation function, and white noise $\xi(t)$.

To incorporate contextual influences (e.g., sensory inputs, task conditions), we employ a hypernetwork (Ha et al., 2016) that maps a context vector $c \in \mathbb{R}^d$, where d is the embedding dimension, to the connectivity matrix:

$$\mathbf{J} = f_h(c), \tag{2}$$

where f_h is a hypernetwork parameterized by Θ_h, yielding context-dependent dynamics. Thus, instead of learning a separate recurrent matrix for each trial, a single hypernetwork can share information across trials to generate appropriate recurrent matrices as needed.

Many combinations of recurrent matrix can reproduce the same dynamics, to reduce the solution degeneracy, we constrain it to low rank $R \leq N$, where R is the user-defined rank (Huang et al., 2024). The recurrent connectivity can then be expressed as :

$$\mathbf{J} = \mathbf{M}\mathbf{N}^{\mathsf{T}}, \tag{3}$$

where $\mathbf{M}, \mathbf{N} \in \mathbb{R}^{N \times R}$. Low-rank matrices constrain the dynamics to evolve along a small number of dominant modes, encouraging simpler and more consistent solutions (Mastrogiuseppe and Ostojic, 2018; Schuessler et al., 2020; Beiran et al., 2021; Dubreuil et al., 2022). In our framework, this low-rank structure is enforced by having the hypernetwork generate the weight factors $\mathbf{M}$ and $\mathbf{N}$. Both the hypernetwork parameters Θ_h and the context variables c are jointly optimized to minimize the mean-squared error between the predicted and recorded neural trajectories. After training, the inferred context for each trial is stored for downstream analysis. When the hypernetwork is unconstrained, we refer to it as HNET-FULL and when constraints are applied, we refer to the resulting model as HNET-LOW.

3. Empirical Results

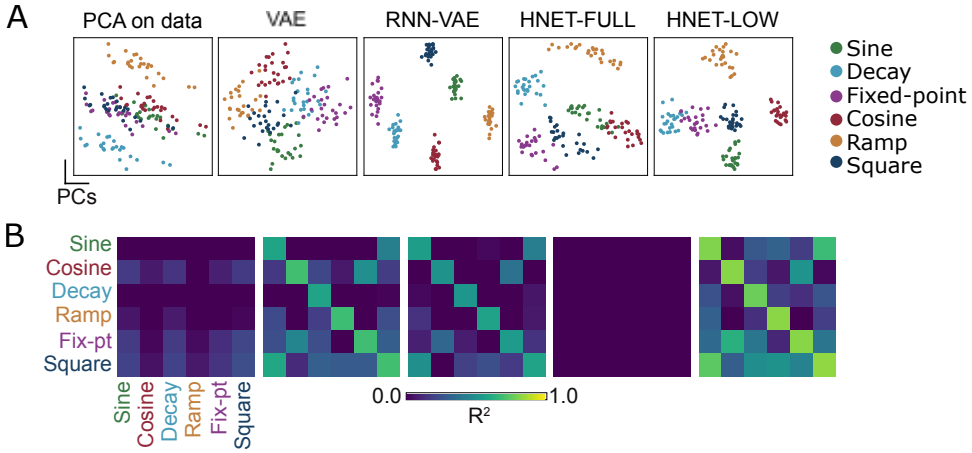

Figure 1: Quantifying the quality of the embeddings. A) 2D PCA visualization of the embeddings obtained using embedding-oriented methods. Each point in the plot corresponds to a trial. B) Confusion matrix of the generalization R^2 scores where rows correspond to the task ID of the mean embeddings and columns indicate the held-out data used for evaluation.

3.1. Low-rank constrained hypernetworks recover task identity and generalize in the student-teacher setup.

To understand the model's behavior in a controlled setting, we evaluated our model by fitting the trajectories collected from the constrained teacher RNN that spanned a diverse range of dynamical regimes. The details of the data generation setup and training is provided in Appendix A. We also validated against a dataset of motor cortical recordings during monkey reaching (Perich et al., 2018). We focused on two model configurations: unconstrained hypernetwork generating full-rank recurrent weights and constrained model generating matrices of fixed rank r. We assessed the quality of embeddings by verifying their separability and generalization, and compared them against the embeddings of other embedding oriented baselines (Fig. 1A): Variational Autoencoder (VAE) and a recurrent VAE (RNN-VAE). We also used Principal Component Analysis (PCA) directly on the data as a classical dimensionality reduction baseline (see Appendix B for more baseline implementations). Note our goal is ultimately to study the structure in the RNN weights, and we use these baselines to verify the success of our learned embeddings.

We trained a Naive Bayes classifier on embeddings from each model to assess if the model learned contexts related to the task. All model embeddings were fully seperable except PCA showing our model performs equally well compared to baselines(see Appendix C). To evaluate generalizability, we performed across-task prediction. For each task, we computed the mean for trained embeddings of all trials and used it to generate predictions

on held-out data from each task. We then measured the accuracy of the predictions using R^2. On synthetic datasets, the low-rank hypernetwork generalized the best as shown by the confusion matrix in Fig. 1B), achieving high R^2. The full-rank model performed worse, suggesting that lack of structural constraints leads to overfitting and poor extrapolation. The generalization and separability benefits of low-rank constraints observed in synthetic data also emerged in single-neuron recordings from monkeys performing a reaching task.

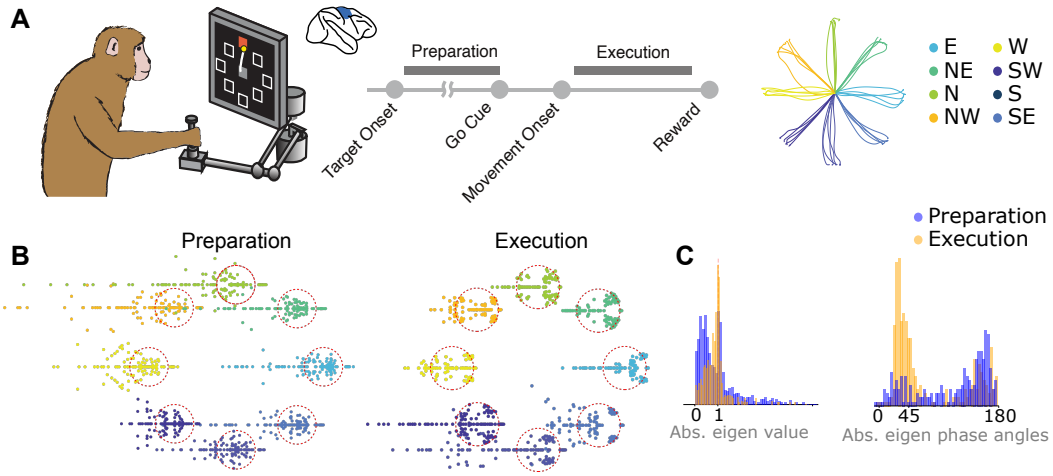

Figure 2: Analysis of trained weights. A) Experimental setup of center-out reaching task. B) Eigenvalue spectrum of trained low-rank recurrent matrices during preparation (left) and execution (right) across 8 directions. During execution, eigenvalues clustered close to the unit circle, indicating dynamics approaching the edge of stability. C) Distribution of absolute of eigen values and angles during preparation and execution of movements.

3.2. Spectral Analysis of recurrent matrix reveals reorganization of dynamics during execution in monkey reaching task.

Next, we analyzed the eigen spectrum of the learned recurrent connectivity to probe dynamical structure across models, tasks, and datasets. In the synthetic dataset, the low-rank hypernetwork produced weights with task-specific eigenspectra with clear structure: oscillatory tasks (sine, cosine) and non-oscillatory tasks (ramp, decay)(see in Appendix C). This task-aligned separation indicates the model captures distinct regimes consistent with input properties. By contrast, the full-rank hypernetwork yielded a dense, isotropic eigenvalue cloud spread across the complex plane, suggesting harder-to-interpret dynamics.

We then analyzed both full and low rank variants of our model trained on the monkey recordings (Fig. 2A). The full-rank model produced diffuse eigenspectra lacking a consistent structure (see Appendix C), while the low-rank variant unveiled clear organization in the eigenspectra. We observed a striking reorganization of the recurrent dynamics across preparation and the execution of movement, in all 8 directions of behavioral contexts as

seen in the Fig. 2B. During preparation, the eigen-spectra was dominated by large negative eigenvalues, indicating strongly contracting, stable geometry, consistent with robust convergence to preparatory states. As the movement began, the spectra reorganized: eigenvalues became more positive, indicating unstable transient dynamics, placing some at the edge of stability(Fig. 2C). This critical regime expanded activity along selected dimensions, enabling reproducible yet flexible trajectories that generate motor output. The system transitioned from a contraction-dominated flow field to one perched at the edge of stability (Fig. 2B(right)), stable enough to avoid chaos, but rich enough to support time-varying execution.

These results show that the low-rank hypernetwork not only generalizes but also exposes meaningful structure in learned dynamics, revealing task-specific features in synthetic settings and conserved motifs in biological data.

4. Discussion

In this work, we introduced a hierarchical model that leverages hypernetworks to parameterize low-rank autonomous RNNs that fit neural data, testing whether task-conditioned embeddings generating recurrent weight matrices could yield interpretable and generalizable models. The proposed approach provided insights into both meta states underlying neural processes and the dynamical regimes of activity. Our findings show that context-conditioned weight generation not only fits neural data but also uncovers meaningful dynamical structure in neural activity, with low-rank constraints serving as inductive biases that separate task-relevant features from noise and enhance generalization.

Acknowledgments

MGP acknowledges support of a Future Leaders award from the Brain Canada Foundation and a J1 Chercheurs-boursiers en intelligence artificielle from the Fonds de recherche du Québec – Santé. GL acknowledges support from NSERC Discovery Grant RGPIN-2018-04821, the Canada Research Chair in Neural Computations and Interfacing, a Canada-CIFAR AI Chair, IVADO, and the Canada First Research Excellence Fund. The authors also acknowledge the support of computational resources provided by Mila, UNIQUE scholarship, the Digital Research Alliance of Canada, and NVIDIA that enabled this research.

References

Manuel Beiran and Ashok Litwin-Kumar. Prediction of neural activity in connectome-constrained recurrent networks. *bioRxiv*, 2024.

Manuel Beiran, Alexis Dubreuil, Adrian Valente, Francesca Mastrogiuseppe, and Srdjan Ostojic. Shaping Dynamics With Multiple Populations in Low-Rank Recurrent Networks. *Neural Computation*, 33(6):1572–1615, 2021.

Fatih Dinc, Adam Shai, Mark Schnitzer, and Hidenori Tanaka. CORNN: Convex optimization of recurrent neural networks for rapid inference of neural dynamics. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Alexis Dubreuil, Adrian Valente, Manuel Beiran, Francesca Mastrogiuseppe, and Srdjan Ostojic. The role of population structure in computations through neural dynamics. *Nature Neuroscience*, 25(6):783–794, 2022.

Lea Duncker and Maneesh Sahani. Dynamics on the manifold: Identifying computational dynamical activity from neural population recordings. *Current opinion in neurobiology*, 70:163–170, 2021.

Daniel Durstewitz, Georgia Koppe, and Max Ingo Thurm. Reconstructing computational system dynamics from neural data with recurrent neural networks. *Nature Reviews Neuroscience*, 24(11):693–710, 2023.

David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

Florian Hess, Zahra Monfared, Manuel Brenner, and Daniel Durstewitz. Generalized teacher forcing for learning chaotic dynamics. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23, 2023.

Ann Huang, Satpreet H Singh, Flavio Martinelli, and Kanaka Rajan. Measuring and controlling solution degeneracy across task-trained recurrent neural networks. *arXiv preprint arXiv:2410.03972*, 2024.

Francesca Mastrogiuseppe and Srdjan Ostojic. Linking connectivity, dynamics, and computations in low-rank recurrent neural networks. *Neuron*, 99(3):609–623.e29, 2018.

Chethan Pandarinath, Daniel J. O'Shea, Jasmine Collins, Rafal Jozefowicz, Sergey D. Stavisky, Jonathan C. Kao, Eric M. Trautmann, Matthew T. Kaufman, Stephen I. Ryu, Leigh R. Hochberg, Jaimie M. Henderson, Krishna V. Shenoy, L. F. Abbott, and David Sussillo. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods*, 15(10):805–815, 2018.

Matthew G Perich, Juan A Gallego, and Lee E Miller. A neural population mechanism for rapid learning. *Neuron*, 100(4):964–976, 2018.

Matthew G. Perich, Charlotte Arlt, Sofia Soares, Megan E. Young, Clayton P. Mosher, Juri Minxha, Eugene Carter, Ueli Rutishauser, Peter H. Rudebeck, Christopher D. Harvey, and Kanaka Rajan. Inferring brain-wide interactions using data-constrained recurrent neural network models. bioRxiv:2020.12.18.423348, 2021.

Kanaka Rajan, LF Abbott, and Haim Sompolinsky. Stimulus-dependent suppression of chaos in recurrent neural networks. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 82(1):011903, 2010.

Friedrich Schuessler, Alexis Dubreuil, Francesca Mastrogiuseppe, Srdjan Ostojic, and Omri Barak. Dynamics of random recurrent networks with correlated low-rank structure. *Physical Review Research*, 2(1):013111, 2020.

David Sussillo and L. F. Abbott. Generating coherent patterns of activity from chaotic neural networks. *Neuron*, 63:544–557, 2009.

Elia Turner, Kabir V Dabholkar, and Omri Barak. Charting and navigating the space of solutions for recurrent neural networks. *Advances in Neural Information Processing Systems*, 34:25320–25333, 2021.

Adrian Valente, Jonathan W Pillow, and Srdjan Ostojic. Extracting computational mechanisms from neural data using low-rank rnns. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Appendix A. Dataset Details

A.1. Teacher Setup for generating synthetic data

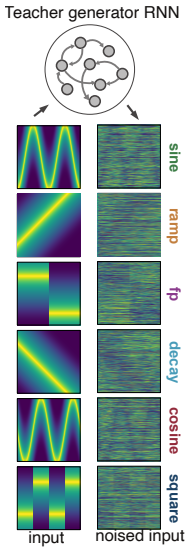

Figure 3: Synthetic data generation process

To evaluate the performance, generalization, and interpretability of our model, we studied a teacher-student paradigm Beiran and Litwin-Kumar (2024). In this work, the teacher is used as proxy for a neural system whose recurrent connectivity structure is known and the student (our model) is trained to mimic the teacher.

We generated synthetic data from a chaotic teacher RNN with predefined recurrent connectivity and input structure. The schematic diagram of the data generation process is shown in the Fig. 3. The RNN was composed of N firing rate neurons, in which the activity $h(t)$ of the neuron i evolved according to the dynamical update:

$$\tau \frac{d\mathbf{h}}{dt} = -\mathbf{h}(t) + g\mathbf{J}\phi(\mathbf{h}(t)) + W_{ext}U(t), \tag{4}$$

where $h(t) \in \mathbb{R}^N$ is the network state at time t, ϕ is nonlinear activation function, Δt is the simulation timestep, τ is the neural time constant, and $u(t) \in \mathbb{R}^N$ is time varying external input. $\mathbf{J} \in \mathbb{R}^{N\times N}$ is the recurrent connectivity matrix sampled from a Gaussian distribution $\mathcal{N}(0, \sigma^2)$. The chaos factor g determines the strength of the recurrent connections, and thus whether $(g > 1)$ or not $(g < 1)$ the network produces spontaneous activity with non-trivial dynamics Rajan et al. (2010). We set $g = 1.8$ to produce chaotic noised versions of the external inputs. $\mathbf{W_{ext}} \in \mathbb{R}^{N\times N}$ is input matrix that maps input signals to the recurrent matrix.

To probe the model's ability to capture different dynamical regimes, we drove the teacher RNN with a range of external input signals $\mathbf{u}(t)$ that shaped the temporal evolution of the RNN. These include oscillatory input patterns (sine, cosine), ramping, decaying, square, and fixed-point (step-like), applied to a subset of neurons (50% of the population) via a sparse input weight matrix $\mathbf{W}_{\text{ext}} \in \mathbb{R}^N$. We refer to different input signal as tasks. We simulated the time series activity of i^{th} neuron for 2 seconds with an additional 0.1 second burn-in period. To simulate the trial-trial variability in the real world, for each input type we generated the activity of the units for different initializations of the input matrix but keeping

the recurrent matrix **J** fixed across seeds and inputs. The resulting activity traces which are essentially the noised external inputs served as ground-truth examples of known dynamical systems with interpretable properties, providing a testbed for assessing the generalization capacity of our model, its ability to infer latent structure, and the interpretability of the weights it generates. In total, we collected synthetic data by driving 6 different external input signals each with $c = 25$ samples of $T = 220$ timesteps.

To probe our model's behavior on data with different intrinsic dimensionalities, we imposed low-rank constraints on the recurrent matrix of the teacher(constrained teacher) and collected the resulting activity data.

We then trained our model to mimic the data generated by the constrained teacher RNN (r=5), that spans a diverse range of dynamical regimes. This allowed us to systematically probe model behavior under controlled conditions, where the underlying dynamics were fully specified and interpretable.

A.2. Motor Cortex Recordings

Two monkeys were trained over several months to perform a two-dimensional center-out reaching task using a planar manipulandum that controlled a cursor on a screen. In each trial, the monkey moved to a central start position, waited through a variable delay, and then reached toward one of eight randomly selected targets arranged uniformly in a circle. A go cue signaled movement initiation, and successful trials required reaching the target within 1 second and holding for 0.5 seconds to receive a reward. We tested the applicability of the proposed approach on this data containing $c = 160$ trials of $T = 150$ timesteps of recordings from $N = 117$ neurons.

Appendix B. Model Architecture and Training details

The latent dimension for all the methods is set to 16.

B.1. VAE

We used a two-layer MLP for both the encoder and decoder. The encoder transforms the flattened input of shape $(T \times features)$ into a latent vector of dimension 16 though hidden layer of 128 hidden units. The decoder mirrors this architecture, decoding from the latent space through a hidden layer of 128 units back to the input dimension.

The reconstruction loss is computed using the reparameterized latent vector, and the total loss includes a standard KL divergence term. During training, we set the batch size to 30 and trained the model for 1500 epochs, selecting the best model using validation loss. We used the Adam optimizer with an initial learning rate of 10^{-3}. Tanh activation was used in the decoder output.

The average training R2 scores achieved on synthetic and monkey reaching task was 0.96$\sim$0.98.

B.2. RNN VAE

For the RNN-based VAE model (RNNVAE), we used a sequence-to-sequence architecture built upon an LSTM encoder and decoder. The encoder comprises a single-layer LSTM

with 32 hidden units, followed by a fully connected layer projecting into a latent space of size 16, where the mean and log-variance are estimated through parallel linear layers. The decoder reconstructs the sequence by transforming the latent vector through a linear projection and passing it through a LSTM layer. The final output is mapped back to the input dimensionality through a fully connected layer. Tanh activation was used in the decoder output. This model operated on a sequence length of 200 time steps.

Total loss was mean squared error between the decoder output and the ground truth trajectory and a standard KL divergence term. During training, we set the batch size to 150 and trained the model for 5000 epochs, selecting used the validation loss. We used the Adam optimizer with learning rate of 10^{-2}. Tanh activation was used in the decoder output.

The average training R2 scores achieved on synthetic and monkey reaching task with RNN-VAE was also 0.96∼0.98.

B.3. Our proposed model

We employed a hierarchical model as shown in the Fig. **??** that employs hypernetwork consisting of a three-layer feedforward neural network that received 16-dimensional context vectors as input and generated the parameters of a downstream recurrent connectivity matrix. The network included a hidden layer of size 128, and ReLU activation was applied at the output. In the full-rank setting, the hypernetwork produced N^2 parameters, where N is the number of neurons, while in the low-rank setting, it generated $2 \times N \times R$ parameters for each trial of activity it was trained, with R denoting a user-defined rank. For synthetic data experiments, we chose $R = 5$, and for monkey reaching data, we used $R = 7$, benchmarked through validation set R^2 scores during training. The hypernetwork weights were initialized with gaussian (mean=0, standard deviation=1) weights and context embeddings were initialized with zeros. We trained the network for 1000 epochs.

Appendix C. Results

C.1. Additional results on synthetic data

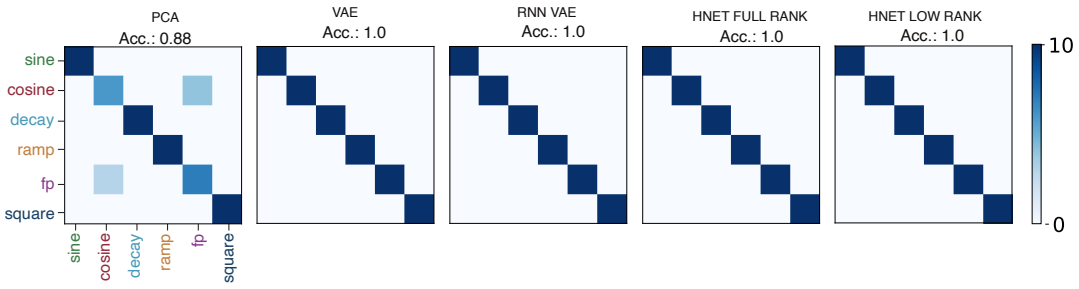

Figure 4: Confusion matrices of task classification accuracies. Rows correspond to the true label of the embedding, and columns are predicted labels. Scale of 10 represents number of test samples.

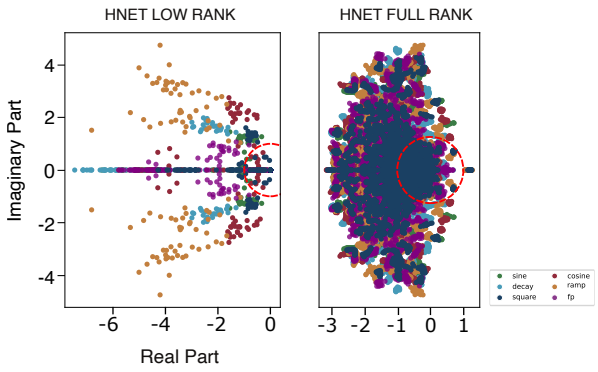

Figure 5: Eigenvalue spectra of learned recurrent weights on synthetic data across tasks. The low-rank hypernetwork reveals task-specific spectral patterns: oscillatory tasks (e.g., sine, cosine) showed eigenvalues with strong imaginary components, while non-oscillatory tasks (e.g., ramp, decay) has real-valued eigenvalues, including deep negatives, indicating diverse decay modes. This alignment with input properties shows the model captures interpretable, task-aligned dynamics. In contrast, the full-rank model produced a dense, isotropic spectrum with less structure, suggesting less stable and less interpretable dynamics that fail to reflect task identity.

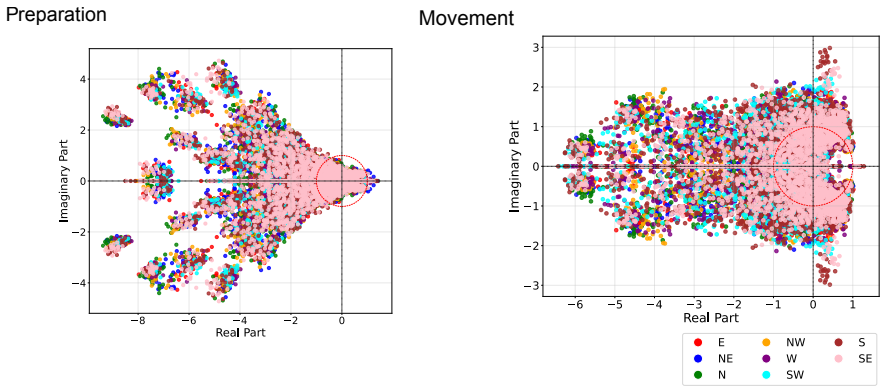

Figure 6: Eigenvalue spectrum of full-rank recurrent connectivity during preparation(left) and execution(right). Each panel shows the eigenvalues of the full-rank recurrent weight matrix, color-coded by reach direction. The red dashed unit circle indicates the stability boundary. During both preparation and execution, the eigenvalues form a dense, cloud-like distribution spread across the complex plane, with many values near or beyond the unit circle. Unlike the structured patterns seen in low-rank models, (like Fig. 2 these spectra are hard to interpret, lacking clear task-specific organization.