Reservoir Computing: An investigation of biophysical neuron modeling

Ashley Chen[1], Jennifer Xia[2], Ananya Yammanuru[2], Nancy M. Amato[2]
[1]University of Minnesota, [2]University of Illinois at Urbana-Champaign

An NSF Expedition in Computing
Mind *in Vitro*

Background

The Mind in Vitro (MiV) research group's overarching goal is to create a computational system based on neural substrates. Our project is an investigation of reservoir computing (RC), a potential algorithm that can harness the properties of neurons.

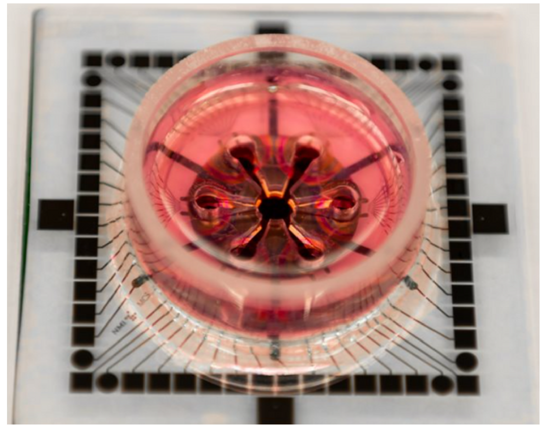

Fig 1: Neuron culture

Introduction

Reservoir computing (RC) is a machine learning architecture that frequently uses RNNs, in which the RNN is fixed and only its readout is trained [1]. RC is most commonly used for temporal classification and prediction tasks [2].

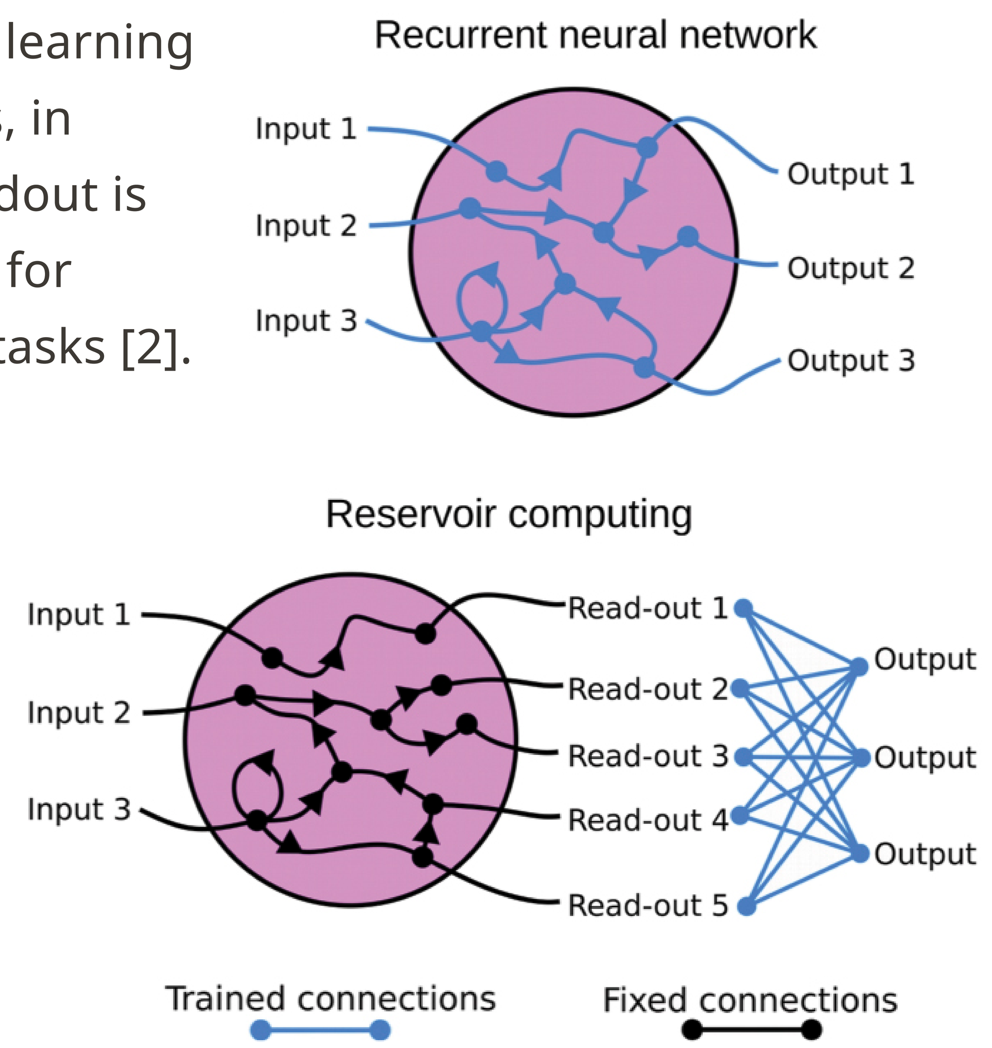

Fig 2: Above: RNN (top) vs RC architecture (bottom)[2]

RC Components:

- Input signal
- Reservoir
  - Untrained network, oftentimes an RNN
  - Weights are typically randomly set and fixed
- Readout
  - Trained on output via ridge regression, MLP, etc.
  - Only layer of RC that is learned

RC Process:

- Input signal is fed into model
- Reservoir projects temporal components of input signal to higher dimensional plane
  - Similar to SVM
- This projection is the readout
- Readout is trained on the target

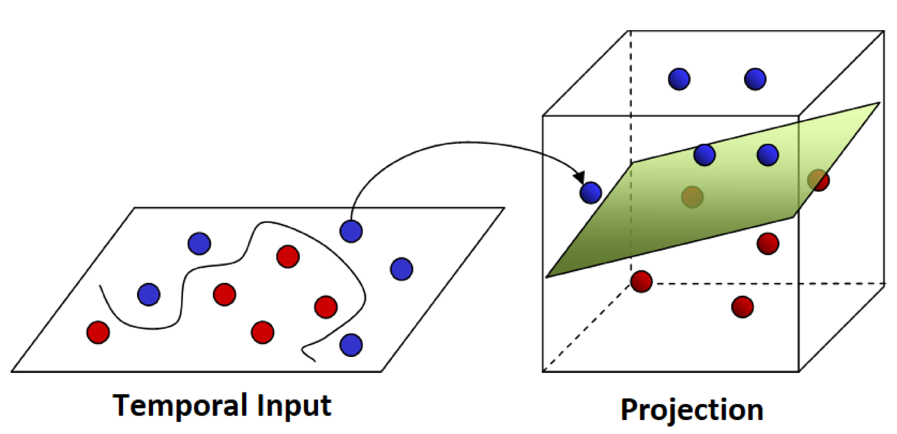

Fig 3: Above: The reservoir projects the input onto a higher dimensional plane, similar to a SVM [3].

Aim

Overall: To assess a potential model for neurological computation.

- Compare baseline ridge regression model to RC model for temporal data
- Test the impact of tuning different RC model parameters
- Prepare to model with MiV simulator, which only accepts spiking data
- Convert real-valued data into spike trains
- Compare spikes for different encoding methods

RC Nengo Model

We used the neurological simulator Nengo to make our reservoir.

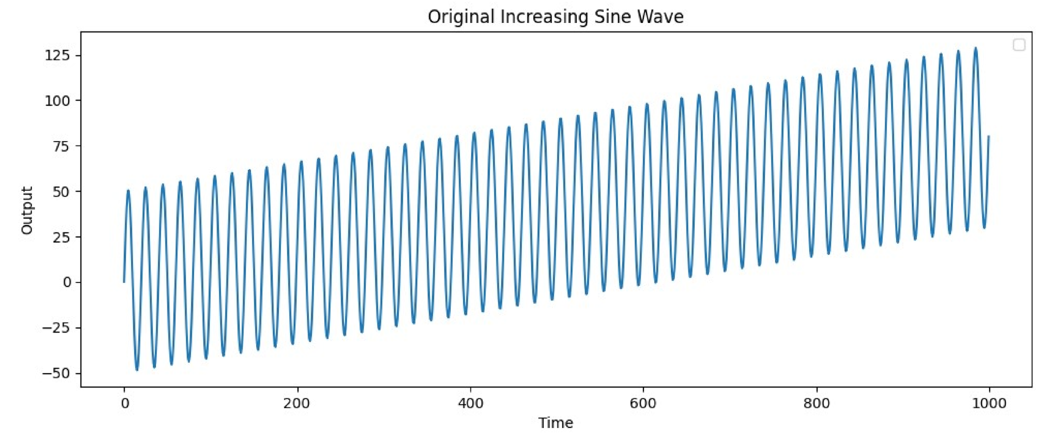

Fig 4: Input signal that we used to train and test the reservoir

We tuned two major reservoir parameters: reservoir size and spectral radius. Reservoir size caused overfitting which made us test regularizing the readout training method as well.

Fig 5: Top: The impact of regularization on preventing overfitting after tuning reservoir parameters. Bottom: Comparing the results of baseline ridge regression (left) with our tuned RC model (right)

Spiking Datasets

We used SNNTorch to convert an increasing sine wave (pictured left) to spikes. The following raster plots represent when a spike occurs in specific neurons in a neural network.

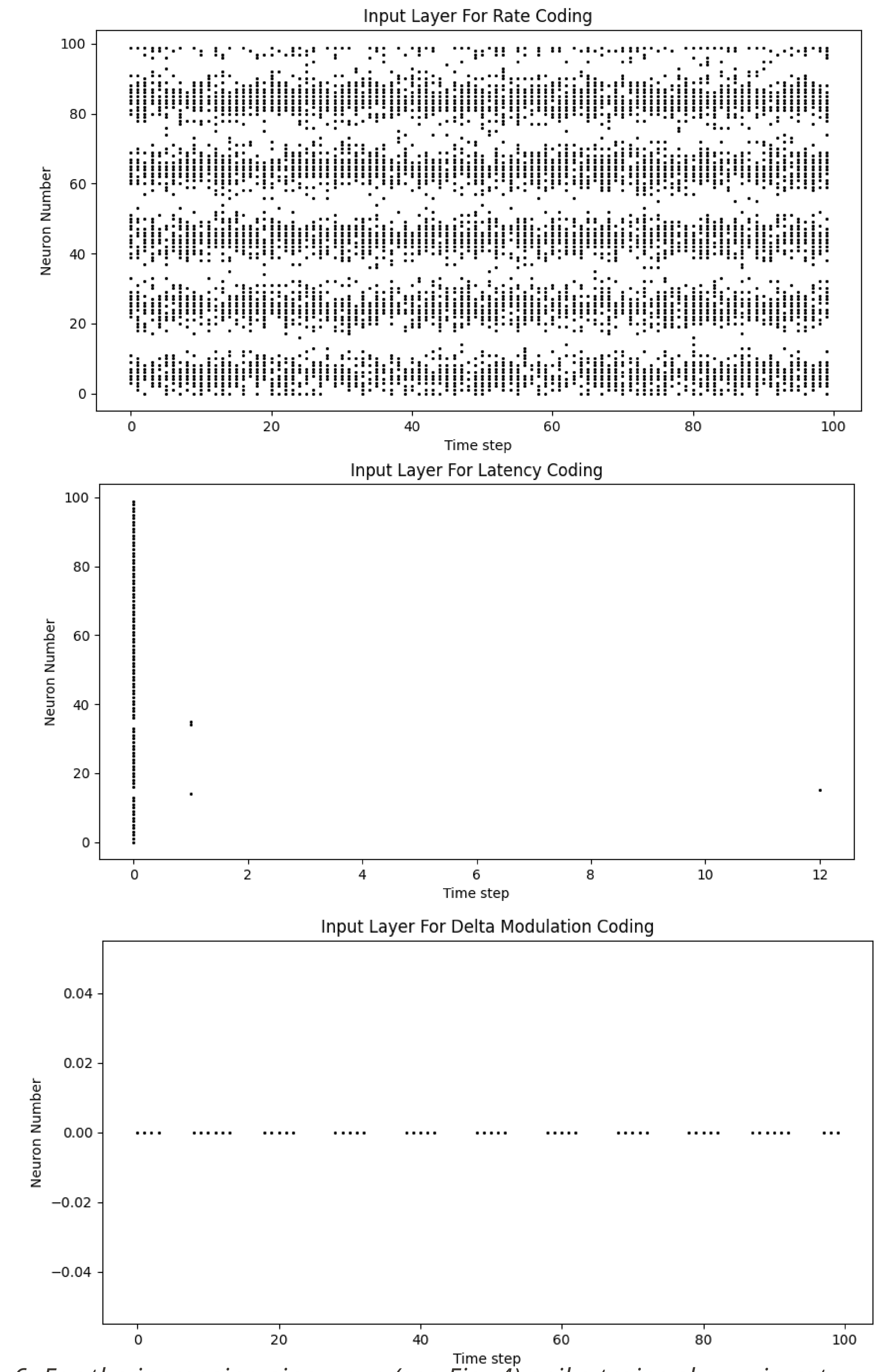

Fig 6: For the increasing sine wave (see Fig. 4) spike trains shown in rate-encoded (top), latency-encoded (middle), and delta-modulated (bottom) strategies.

Conclusion and Future Work

With some parameter tuning, our model is better at training the temporal elements of a signal than baseline ridge regression.

Next steps:

- Continue investigating parameters and architectures for Nengo RC model
- Test Nengo RC model on spiking datasets
- Use MiV simulator to implement RC model and compare this with Nengo model

Acknowledgements

We would like to thank Ananya Yammanuru, Nancy Amato, Marco Morales, Courtney McBeth, Rachel Ann Moan, Seung Hyun Kim, Frithjof Gressmann, Simon Kato, Parasol Lab, CS-SRP, IGB, and CRA-WP.

References

[1] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," Computer Science Review, vol. 3, no. 3, pp. 127–149, 2009. doi:10.1016/j.cosrev.2009.03.005
[2] Matteo Cucchi et al. "Hands-on Reservoir Computing: A Tutorial for Practical Implementation." IOPScience, 2022, iopscience.iop.org/article/10.1088/2634-4386/ac7db7/meta.
[3] N.J, Nalini & Palanivel, s. (2015). Music emotion recognition: The combined evidence of MFCC and residual phase. Egyptian Informatics Journal. 17. 10.1016/j.eij.2015.05.004.