# Incorporating an adaptive learning rate in a neural model of action selection

Sverrir Thorgeirsson, Brent Komer, Chris Eliasmith {sverrir.thorgeirsson, bjkomer, celiasmith}@uwaterloo.ca
Centre for Theoretical Neuroscience, University of Waterloo <http://ctn.uwaterloo.ca>

University of Waterloo · CTN Waterloo · CNRG computational neuroscience research group compneuro.uwaterloo.ca

## Abstract

In previous work, we implemented spiking neuron models that use a biologically realistic action selection system to solve complex cognitive tasks. However, such models require the fine-tuning of multiple parameters to reach a desired level of performance. Recently, we demonstrated that a local, online learning rule, which only requires a single parameter, the learning rate, is sufficient for teaching our model how to solve a general cognitive sequencing task. Here, we refine our method by showing that adding adaptive learning is more robust regarding our choice of parameters, and will achieve better performance on all versions of the cognitive task that we tested. These results provide a foundation for building complex cognitive models that require no hand-tuning of parameters.

## Motivation

- Spiking neural models that solve cognitive tasks require many parameters.
- These parameters can be learned in an online fashion, but the learning rate itself still needs to be chosen.
- Learning rate has a large impact on model performance and the optimal value is different for different tasks.
- There is evidence that the brain contains mechanisms for modulating its learning rate during task performance.

## Adaptive Learning Rate Algorithm

The standard method for online learning in our models is to use a delta rule to modify the connection weights:

$$\Delta\omega_{i,j} = \alpha x_i (t_j - y_j)$$

We use the RMSProp algorithm to adapt the learning rate over time. This works by dividing an initial learning parameter by the root of the exponentially decaying average of squared gradients from previous timesteps:

$$\sqrt{\sum_{i=1}^{t} g_i^2 \cdot a^{t+1-i} + \varepsilon}$$

## Cognitive Task

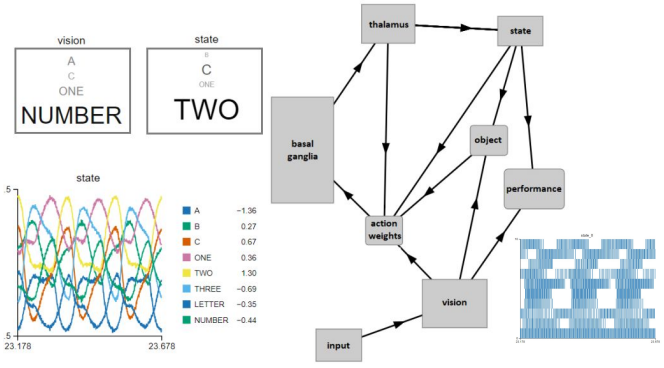
- Simple cognitive task for testing the learning approach.
- The model must recite a sequence of symbols for a given alphabet.
- Expected behaviour can be modelled as a production system of if->then rules.
- The model is Implemented by neurons computing utility functions for each action.

| IF | THEN |
|---|---|
| $v$ = LETTER AND $m$ =A | $m$ = B |
| $v$ = LETTER AND $m$ =B | $m$ = C |
| $v$ = LETTER AND $m$ =C | $m$ = A |
| $v$ = NUMBER AND $m$ =ONE | $m$ = TWO |
| $v$ = NUMBER AND $m$ =TWO | $m$ = THREE |
| $v$ = NUMBER AND $m$ =THREE | $m$ = ONE |

$$U_1 = \alpha_1 \cdot v \cdot \text{LETTER} + \beta_1 \cdot m \cdot A$$
$$U_2 = \alpha_2 \cdot v \cdot \text{LETTER} + \beta_2 \cdot m \cdot B$$
$$\ldots$$
$$U_6 = \alpha_6 \cdot v \cdot \text{NUMBER} + \beta_6 \cdot m \cdot C$$

## Neural Model

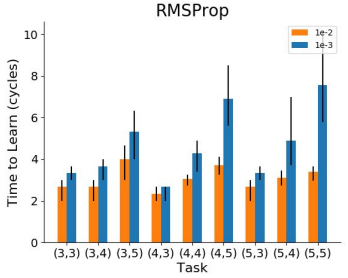
We use the Neural Engineering Framework and the Semantic Pointer Architecture to construct a model that performs the task using spiking neurons. To perform action sequencing, we use a neural implementation of the cortex-basal ganglia-thalamus system. The model consists of visual information, a working memory, the state of the motor system, and other critical components of the human neural action selection system. To compute the utility of actions, we use the connections between the cortex and the striatum in the basal ganglia. The basal ganglia will then select the largest of those utility values, and the corresponding action will be selected via connections from the basal ganglia to the thalamus.



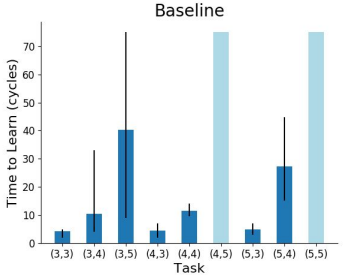

Screenshot of the graphical user interface with the Nengo Model

## Results

Each task is defined by a pair of numbers (n, m) corresponding to the number of symbols in each alphabet and the number of alphabets. Performance is evaluated based on how long the model takes to consecutively complete every transition in every alphabet.




For the RMSProp algorithm, setting $\alpha$ to $10^{-2}$ and $10^{-3}$ gave equivalent or better results than the baseline for any value of $\alpha$ or any version of the task.




The baseline algorithm uses a spiking delta learning rule and is depicted above using $\alpha = 10^{-3}$.

## Summary

- Significantly better performance than the baseline
- Wider range of initial learning rates perform well
- Robust to different versions of the sequencing tasks