Astrocyte Modulated Synaptic Plasticity in LSMs

Astrocyte Modulated Synaptic Plasticity in Spike Timing Dependent Plasticity to improve performance of Liquid State Machines

Nivedya S Nambiar, 190070039
Guided by
Prof. Udayan Ganguly
Anmol Biswas

Report for BTech Project - 1, Autumn 2022

Department of Electrical Engineering
Indian Institute of Technology Bombay

Contents

1 Acknowledgements

I would like to thank my project guides, Anmol Biswas and Prof. Udayan Ganguly for their suggestions and insights throughout the course of the project, enriching the learning experience, and for providing constant support without which the project could not have been completed. I would also like to thank the MeLoDe Algorithms team whose inputs at weekly interactions enabled me to revise my perspective, while giving me a platform to share my ideas, gain feedback, and clarify doubts. I would also like to thank the Department of Electrical Engineering for granting me the opportunity to work on this project and for providing me with access to resources that were imperative in completing this project.

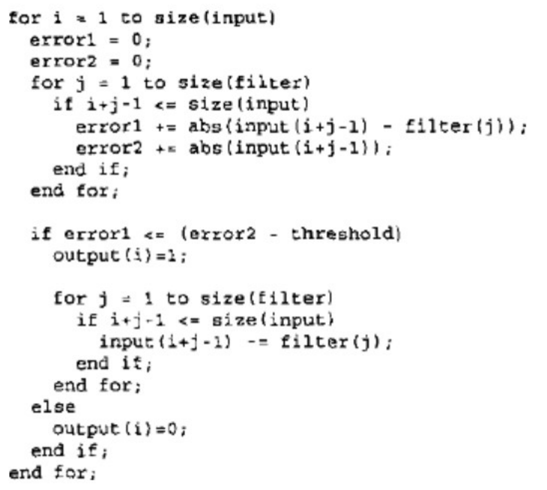
2 Introduction

The idea of astrocyte modulation of synaptic plasticity is borrowed from knowledge of biological neuronal networks that have special cells called astrocytes that attune the operation of brain networks but do not participate directly in information transfer like neurons. This has been extended to liquid state machines as described in [2]. This project aims to implement this idea of astrocyte modulation as described in [2] for the purpose of speech recognition. First the framework of NALSM (neuron-astrocyte liquid state machine) was used for recognition of speech commands in the Google mini speech commands dataset, and for a smaller version of the TI-46 dataset.
To convert the audio inputs into spikes in time, an implementation of the Lyon ear model was used followed by BSA encoding to get spike patterns for each sample. Following this an existing implementation of NALSM was employed for classification.
The NALSM implementation was then tested for the entire TI-46 dataset with 19038 samples, which had the spikes stored in dense matrices for each sample. However, the performance was not satisfactory initially, hence the functions implementing iteration through the LSM and application of STDP were rewritten. The results for the same are presented in following sections. A final accuracy of approximately 90.23% was obtained in the train dataset and 86.70% on the test dataset.

3 Lyon Ear Model and BSA Encoding in Python

The Lyon ear model is an auditory model shows the propagation of sound in the inner ear and the conversion of sound into neural representation. This model is implemented in Python in the package lyon, as described by authors in [5]. The model combines a series of filters that recreate the traveling pressure waves with Half Wave Rectifiers (HWR) to detect the energy in the signal and several stages of Automatic Gain Control (AGC) [5]. The result is a cochleogram which is a two-dimensional time-frequency representation that showcases the spectral information in the sample.

To convert this cochleogram into spikes per channel versus time, the BSA (Bens Spiker Algorithm) is used. This encoding algorithm was introduced in [4]. The pseudocode for the algorithm is represented in the figure below, taken from the same paper. The implementation of the algorithm in Python was available in the package pyspikes [1].



```
for i = 1 to size(input)
  error1 = 0;
  error2 = 0;
  for j = 1 to size(filter)
    if i+j-1 <= size(input)
      error1 += abs(input(i+j-1) - filter(j));
      error2 += abs(input(i+j-1));
    end if;
  end for;

  if error1 <= (error2 - threshold)
    output(i)=1;

    for j = 1 to size(filter)
      if i+j-1 <= size(input)
        input(i+j-1) -= filter(j);
      end if;
    end for;
  else
    output(i)=0;
  end if;
end for;
```

Figure 1: BSA Pseudocode [4]

The Lyon ear model followed up by the BSA encoding scheme converts the audio input into spikes in time which can be now fed into a liquid state machine.

4 Performance on Google Mini Speech Commands Dataset

The performance of the network with STDP on the Google Mini Speech commands dataset containing 8 classes was tested. The resulting train score was obtained as 0.93859375 and the test score was 0.671875. The resulting confusion matrix for the test dataset is as shown in the figure below. The code is documented at `https://colab.research.google.com/drive/1d284PD2cQ3U0t6W-47jD3urREPoj-1H1?usp=sharing`

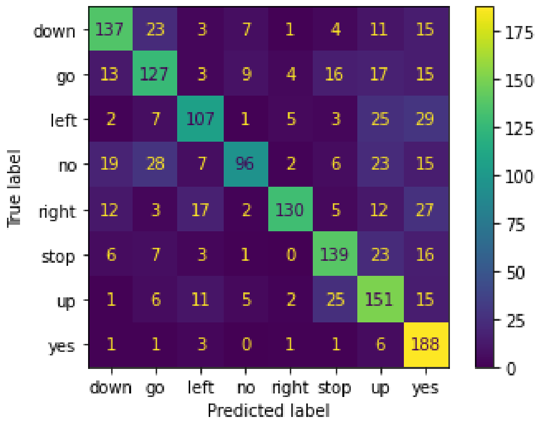


Figure 2: Confusion matrix for Google mini speech commands dataset

5 Code refactoring and performance on TI-46 dataset

Initially when the existing functions for LSM was extended to the TI-46 dataset with 46 classes and 19038 samples, the performance was not satisfactory - with a train score of 59.52% and a test score of 49.63%. For this dataset, the spikes were pre-generated in the .mat file and organised in dense matrices.

The existing code for the LSM function and STDP implementation were refactored and the network parameters were tuned to improve the LSM performance.

5.1 Refactoring the code for State Update in the Liquid State Machine

Based on the description of differential equations in [2] we refactor the function run_LSM which is responsible for updating the state, i.e., computing the membrane potential, input synaptic currents to the neurons in the liquid, and recording the spiking in the reservoir.

5.2 Tuning Network Parameters

As described in [2], the reservoir aids in better classification when the input spiking rates and the spiking rate of the liquid are roughly equal, i.e., when the liquid operates at what is referred to as the "edge of chaos", hence we tune the LSM network parameters like the value of the initial weights, the input connection density, and the value of λ, the horizontal shift featuring in the probability of connection between any two neurons in the liquid. The LSM without STDP implementation was trained using a linear classifier for the output layer. For LqW=2, in_conn_density=0.08, inh_fr=0.35, $\lambda = 3$, an train score of 0.8958702645919506 and a test score of 0.8652482269503546 were obtained. The input spiking was 0.01893004968468113 and the LSM spiking was 0.014724582796800416 for this choice of parameters. Next the implementation of STDP was refactored.

5.3 Refactoring STDP implementation

The implementation of STDP as described in [2] was used to refactor the existing implementation of STDP. Initially a train score = 0.8578556890552164 and a test score = 0.7583399001838718 were obtained. Subsequently, the update of the astrocyte parameter A^-_{astro} was modified to normalise the number of spikes in the liquid and the input layer by the number of neurons in the liquid and the input layer respectively according to the following differential equation, where A^-_{astro} maps to the depression rate A^- and b_{astro} maps to the potentiation rate A^+.

$$\tau_{astro}\frac{dA^-_{astro}}{dt} = -A^-_{astro} + w_{astro}(\frac{1}{N_{liq}})\Sigma_{i\in N_{liq}}\delta(t-t_i) - w_{astro}\frac{1}{N_{inp}}\Sigma_{j\in N_{inp}}\delta(t-t_j) + b_{astro}$$

As a result the accuracy increased, and now the train score obtained was 0.8688201693913729 with a test score of 0.7722616233254531. Additionally the time scale of the update of A_{astro}, i.e. τ_{astro} was decreased from 10 to 5, to aid in faster update of A_{astro}. As a result the train score was 0.8847088175431685 with a test score of 0.7848699763593381.
The weight w_{astro} was modulated in addition, however yielded only marginally higher performance. Increasing the number of epochs also could not boost the performance significantly. The reservoir was observed to be overfitting, in a way. Therefore another possibility of modulating the network was considered as described next.
In the implementation, there was no update of b_{astro}, the potentiation rate. However, since the paper [2] mentioned a decay in b_{astro} at a rate of 0.99, this possibility of modulating the potentiation rate was also explored to test out its influence on the performance of the reservoir. Various update types for $A^+ = b_{astro}$ were experimented, and the results are shown in the table below. The given results are for 1 epoch of STDP. An increment in the A^+ value however yielded undesirable results where the spiking in the reservoir was deactivated entirely.

Update equation	Input Spiking (test)	LSM spiking (test)	Train score	Test score
$A^+ = 0.99A^+$	0.018846	0.014820	0.87184	0.8308379
$A^+ = A^+(1 - 1/9999)$	0.018846	0.012380	0.898036	0.85789
$A^+ = A^+(1 - 1/10^6)$	0.01865247	0.009429	0.904208	0.85106
$A^+ = A^+(1 - A^+/10^5)$	0.0186524	0.009527	0.900269	0.854741
$A^+ = A^+(1 - (A^+)^2/10^5)$	0.0186524	0.009489	0.9102488	0.858156

It can be observed that increasing the nonlinearity in the update equation for

A^+ increased the accuracy for the train and test datasets. Hence the final update equation for $b_{astro} = A^+$ is as follows.

$$b_{astro} = b_{astro}(1 - \frac{(b_{astro})^2}{10^5})$$

The resulting confusion matrix after implementing STDP on the NALSM with the final choice of update equation for A^+ is given below.

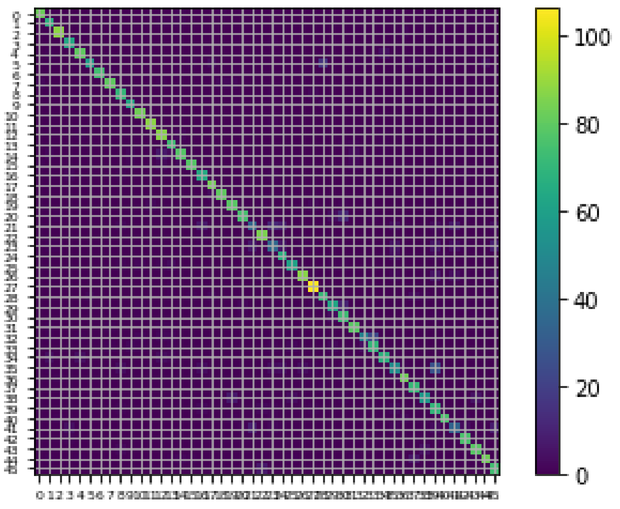


Figure 3: Confusion matrix for the TI-46 test dataset

The value of A^-_{astro} after STDP was obtained as 0.14934722823122565 where the starting value was 0.15 while that for b_{astro} was 0.14948961660780127 with the same starting value of 0.15.

6 Experimenting with MNIST-784 dataset for the same NALSM+STDP framework

The refactored code was employed for the MNIST-784 dataset to test its generalisability. As a result the train score obtained was 0.97556 and the test score was 0.9344. The resulting confusion matrix is shown below.

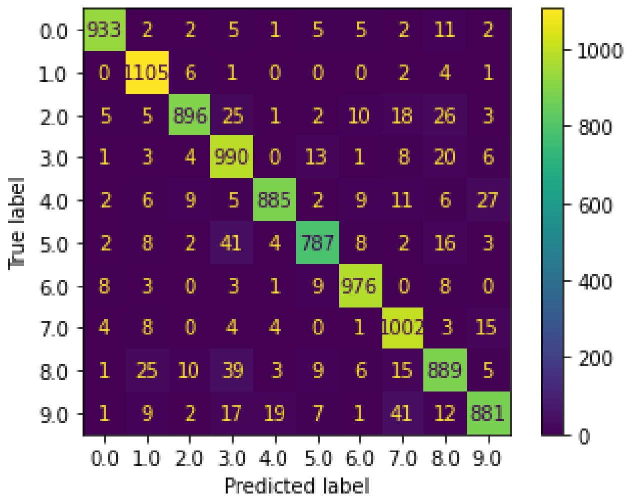


Figure 4: Confusion matrix for MNIST-784 testing

7 Conclusion

The key observation in the experiments with STDP was that any scheme that speeds up the decrease in A^-_{astro}, like decreasing the time scale τ_{astro} of decay of A^-_{astro}, or including a decay in the potentiation rate $A^+ = b_{astro}$ which in addition to decreasing the value of A^-_{astro}, tunes down the weight increase for causal STDP. In effect, the magnitude of updates due to STDP and anti-STDP are decreased successively for every passing time step in the sample, slowing down the weight change for all synapses. This could be viewed as a depression of sorts that regulates the action of the long term potentiation, therefore reinforcing that a combination of short-term depression and long-term potentiation could potentially improve performance of neural networks relying on spike timing dependent plasticity for weight update.

8 Future Work

The exact mechanism of how the accuracy was increased by these updates needs to be explored by graphical visualisation of the results and the dynamics of the reservoir neurons. This would help conclude if any short term depression indeed operates to regulate the action of long term potentiation.
To further test the hypothesis that short term depression and long term potentiation could work together with their complementary actions to build LSMs capable of higher accuracies, an alternative STDP rule namely fatiguing STDP could be explored[3]. The FSTDP (fatiguing STDP) rule aims to decrease the probability of potentiation caused by spike-timing coincidences resulting from

high rates. This is implemented by scaling the effect of presynaptic spike on synaptic weight by a factor depending on the normalised coordinated firing rates instead of the absolute coordinated firing rates that are biased to high spike rates.

9 Code links

Code for refactored code and testing done for TI-46 dataset - https://colab.research.google.com/drive/1yuPJn_CZTJtRGs2duJ1VF71Z3oCDeH82?usp=sharing
Code for MNIST-784 dataset utilising the refactored code - https://colab.research.google.com/drive/1qrsK2kfm8xyaErMLGRC3AITvnPS9fbAz?usp=sharing

References

[1] Akshay Raj Gollahalli. Spike Encoders. https://github.com/akshaybabloo/Spikes, 2020.

[2] Vladimir A. Ivanov and Konstantinos P. Michmizos. Increasing liquid state machine performance with edge-of-chaos dynamics organized by astrocyte-modulated plasticity. *CoRR*, abs/2111.01760, 2021.

[3] Timoleon Moraitis, Abu Sebastian, and Evangelos Eleftheriou. The role of short-term plasticity in neuromorphic learning: Learning from the timing of rate-varying events with fatiguing spike-timing-dependent plasticity. *IEEE Nanotechnology Magazine*, 12(3):45–53, 2018.

[4] B. Schrauwen and J. Van Campenhout. Bsa, a fast and accurate spike train encoding scheme. In *Proceedings of the International Joint Conference on Neural Networks, 2003.*, volume 4, pages 2825–2830 vol.4, 2003.

[5] Sciforce. Our Adaptation of Lyon's Auditory Model for Python. https://medium.com/sciforce/our-adaptation-of-lyons-auditory-model-for-python-4f41adf55d4e, 2019.