Studying effect of STDP on LSM topology

Mihir Ghumbre[1], Siddhant Pravin Shingade[1], Mayur Ware[1]
[1]*Indian Institute of Technology Bombay, Powai, Maharashtra, India - 400076*
Email: mihirghumbre@ee.iitb.ac.in, siddhantshingade@ee.iitb.ac.in, mayurware@ee.iitb.ac.in

Introduction
The learning capability of an information processing device is its most delicate aspect, and that the availability of sufficiently many training examples is a primary bottleneck for goal-directed (i.e, supervised or reward-based) learning.
Liquid State Machines (LSM)'s architecture is designed to make the learning as fast and robust as possible. It delegates the primary load of goal-directed learning to a single and seemingly trivial stage: the output- or readout stage ,which typically is a very simple computational component.

Methodology
LSM can be thought of as a map from an input stream of data u(.) to an output stream y(.). It consists of an input layer ,a complex network of recurrently connected neurons and an output layer. A linear classifier connects the LSM network with the output layer. Learning happens at this stage i.e. the weights at this stage are trained; rest other weights are pre defined. Therefore the LSM acts as a pre-processor for the output layer. This same LSM network can therefore serve as a pre-processor for many output neurons with each output neuron extracting and learning different types of information for serving a variety of purposes. Every neuron in LSM is typically modeled as a leaky integrate and fire neuron.
A simple architecture of LSM is shown in Fig. 1 and Fig. 2. $X^M(t)$ refers to the input at a time t. It can also be viewed as the state of the LSM that is visible to the output at given time t. The output y(t) is a learning function of this input.
The advantage of using a LSM architecture is that, since the learning happens only at the output stage , it is robust and effective. Also this architecture can be generalized to pre-process a variety of new input streams.

Results and Conclusions
We tested the code for implementation of LSM using the given dataset. Fig. 3 shows the graph of the Presynaptic membrane potential whereas, Fig. 4 shows the graph of the Postsynaptic spikes generated after successful implementation.
Fig. 5 shows the accuracy of the model with and without LSM. We can therefore conclude that the use of LSM architecture improves the accuracy of the classification.

Stage 2 Steps
In the next part of this project we will update and modify the existing code to include the effects of Spike Time Dependent Plasticity (STDP). We would use the learning algorithm referred earlier to do the same. The weights updated by this process will make the learning as fast and robust as possible. If the presynaptic neuron fires a spike before a postsynaptic neuron, the connection between the two neurons strengthens. Conversely, if a presynaptic neuron spikes after a postsynaptic neuron, there is no correlation between the two neurons. The connection weights between neurons are weakened.

$$\Delta W = \sum_{t_{pre}} \sum_{t_{post}} W(t_{pre} - t_{post})$$

In the above formula, the ΔW of synaptic weight change is the sum of all presynaptic spike time and postsynaptic spike time on the function W, and the W is defined by

$$W(\Delta t) = \begin{cases} A_{pre} e^{-\Delta t/\tau_{pre}} & \Delta t > 0 \\ A_{post} e^{-\Delta t/\tau_{post}} & \Delta t < 0 \end{cases}$$

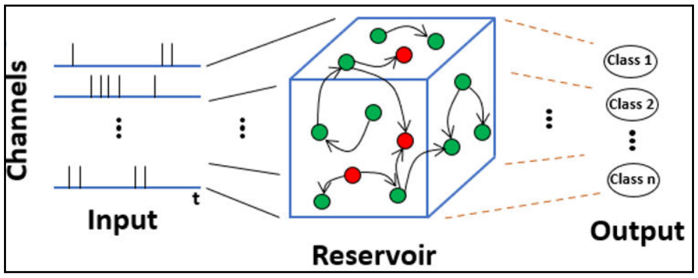


Fig. 1 LSM Architecture – spiking input channels, randomly connected reservoir of excitatory (green) and inhibitory (red) neurons and output neurons, the input and reservoir weights are fixed while the classifier weights are trained

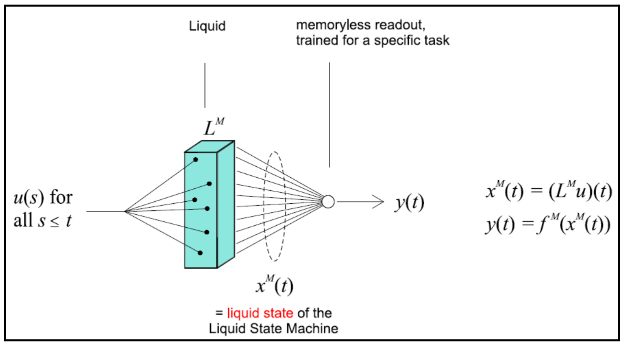
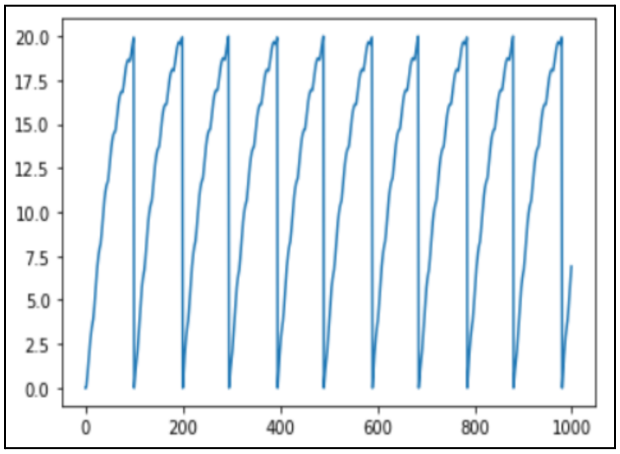

Liquid
memoryless readout, trained for a specific task
L^M
$u(s)$ for all $s \leq t$
$y(t)$
$x^M(t) = (L^M u)(t)$
$y(t) = f^M(x^M(t))$
$x^M(t)$
= liquid state of the Liquid State Machine

Fig. 2 Structure of a Liquid State Machine (LSM) M, which transforms input streams u(·) into output streams y(·).



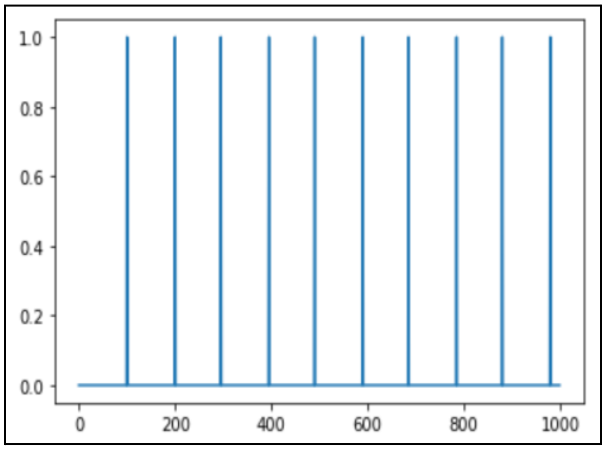
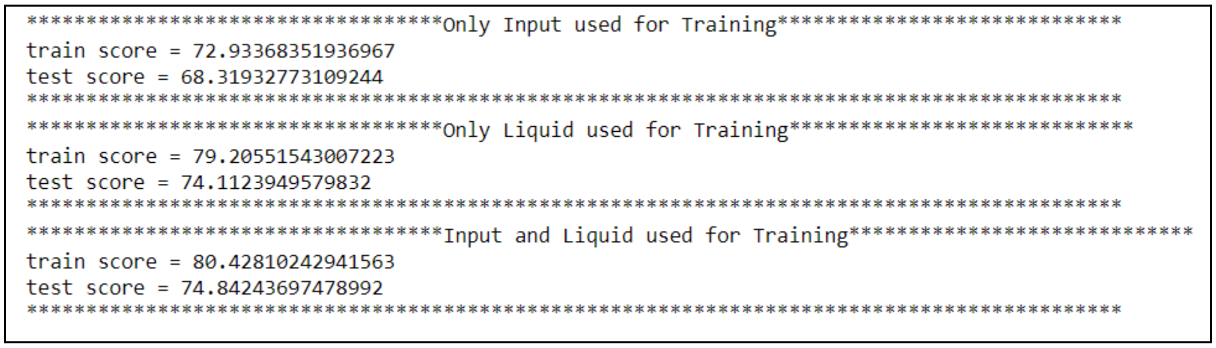


Fig. 3 Presynaptic Membrane Potential

Fig. 4 Spike Train in Postsynaptic neuron



```
***********************************Only Input used for Training*****************************
train score = 72.93368351936967
test score = 68.31932773109244
*********************************************************************************************
***********************************Only Liquid used for Training*****************************
train score = 79.20551543007223
test score = 74.1123949579832
*********************************************************************************************
***********************************Input and Liquid used for Training****************************
train score = 80.42810242941563
test score = 74.84243697478992
*********************************************************************************************
```

Fig. 5 Comparing accuracy of the model

References

[1] *A. Gorad, V. Saraswat, U. Gangulyl., Predicting Performance using Approximate State Space Model for Liquid State Machines, 1901.06240v1 [cs.NE] 18 Jan 2019.*
[2] *Vivek Saraswat, Ajinkya Gorad, Anand Naik, Aakash Patil and Udayan Ganguly, 2021 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-3900-8/21, (2021).*
[3] *Wolfgang Maass, Liquid State Machines: Motivation, Theory, and Applications, Ch. 1, World Scientific Review Volume 189˙v2, (2010).*