Liquid State Machine for CIFAR-10 Classification

Rishabh Ramteke
170070046
Indian Institute of Technology Bombay
Mumbai, India
170070046@iitb.ac.in

Udayan Ganguly
(Guide)
Indian Institute of Technology Bombay
Mumbai, India
udayan@ee.iitb.ac.in

***Abstract*—Liquid State Machine (LSM) is a neural model and a type of reservoir computer that uses a spiking neural network with real time computations which transforms the time varying inputs stream to a higher dimensional space. The concept of LSM is a novel field of research in biological inspired computation with most research effort on training the model as well as finding the optimum learning method. CIFAR-10 is an established computer-vision dataset used for object recognition and is widely used for testing classification algorithms. In this thesis project we have used LSM for image classification on CIFAR-10 dataset. The main goal was to decrease the computation time and increase the accuracy. In LSM, preprocessing of data plays an important role and thus I investigated the performance of LSM on converting the image from spatial domain to frequency domain.**

***Index Terms*—Liquid state machine, Short term plasticity, Reservoir computing, Spiking neural network**

I. Introduction

Spiking neural networks (SNNs) are artificial neural networks that closely imitate natural neural networks. SNNs not only incorporate neuronal and synaptic state, but also the concept of time into their operating model. The neurons in the SNN do not transmit information at each propagation cycle (which generally happens with typical multi-layer perceptron networks), but instead transmit information only when the membrane potential reaches a specific threshold value. When the membrane potential reaches the threshold value, the neuron fires and generates a signal that travels to other neurons which, in turn, increase or decrease their potentials in response to this signal.

The most prominent spiking neuron model is the leaky integrate-and-fire model. In this, the momentary activation level is normally considered to be the neuron's state, with incoming spikes pushing this value higher or lower, until the state eventually either decays or if the firing threshold is reached, the neuron fires. After firing the state variable is reset to a lower value. According to (Wall and Glackin, 2013), SNN is a third generation artificial neuron that is most biologically-inspired. SNNs are becoming a dominant agent for brain-inspired neuromorphic computing - emulating the brain with computational hardware (Sharbati et al., 2018).

Reservoir computing is a framework for computation derived from recurrent neural network (RNN) theory that maps input signals into higher dimensional computational spaces through the dynamics of a fixed, non-linear system called a reservoir (Tanaka et al., 2019). These type of algorithms with shallow networks are computationally cheaper since only the readout layer is trained as the reservoir dynamics are fixed. By reducing network depth and plasticity, reservoir computing minimizes computational power and complexity, making the algorithms optimal for edge devices. However, as a trade-off for their frugal nature, reservoir computing sacrifices computational power compared to state-of-the-art methods.

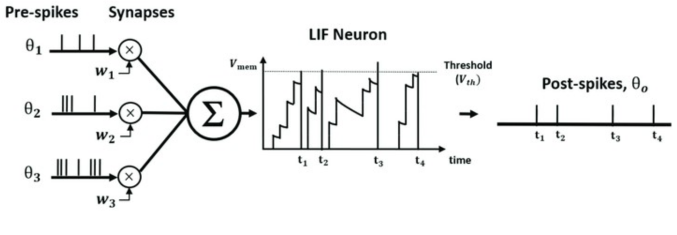


Fig. 1. Illustration of Leaky Integrate and Fire (LIF) neuron dynamics (Lee et al., 2020)

A liquid state machine (LSM) is a type of reservoir computational model that uses a spiking neural network. The nodes are randomly connected to each other. The recurrent nature of the connections turns the time varying input into a spatio-temporal pattern of activations in the network nodes. The spatio-temporal patterns of activation are read out by linear discriminant units.

According to (Maass et al., 2002), LSM model was developed from the viewpoint of computational neuroscience. Its main benefit is that it performs real time computations which transforms the time varying inputs stream to a higher dimensional space. It has three main components:

(i) Input Layer
(ii) Reservoir or Liquid
(iii) Memoryless readout layer

The input to the LSM is of the form of spike trains. Various decoding methods exist for interpreting the outgoing spike train as a real-value number, relying on either the frequency of spikes (rate-code), the time-to-first-spike after stimulation, or the interval between spikes.

In Section-II consists of literature review of various research papers related to the thesis project. Section-III describes the project idea. Section-V describes preprocessing of data. We conclude the paper with our analysis in Section-VI.

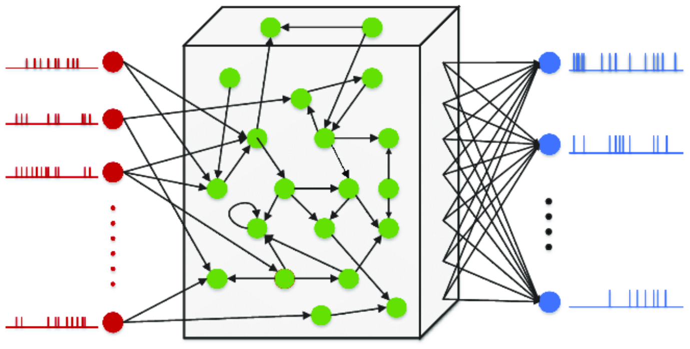


Fig. 2. A model of the liquid state machine (Liu et al., 2019)

II. Related Works

A. SpiLinC: Spiking Liquid-Ensemble Computing for Unsupervised Speech and Image Recognition (Srinivasan et al., 2018)

In this research paper, the authors proposed an SNN consisting of input neurons sparsely connected by plastic synapses to a randomly interlinked liquid, referred to as Liquid-SNN, for unsupervised speech and image recognition. It adapts the strength of the synapses interconnecting the input and liquid using Spike Timing Dependent Plasticity (STDP), which enables the neurons to self-learn a general representation of unique classes of input patterns. This infer the class of a test input directly using the liquid neuronal spiking Activity. STDP postulates that the strength (or weight) of a synapse depends on the degree of timing correlation between the corresponding pre- and post-neuronal spikes. The authors used the power-law weight-dependent STDP model which is described as:

$$\Delta w = \eta \times [e^{-\frac{t_{post} - t_{pre}}{\tau}} - STDP_{offset}] \times [w_{max} - w]^{\mu} \quad (1)$$

where Δw is the change in the synaptic weight, η is the learning rate, t_{pre} and t_{post} are respectively the time instants of a pair of pre- and post-neuronal spikes, τ is the STDP time constant, w_{max} is the maximum bound imposed on the synaptic weight, and w is the current weight. If the pre-synaptic neuron fires first, then the weight is increased; if the post-synaptic neuron fires first then the weight is decreased.

Standard LSM have fixed synaptic connections between the input and liquid followed by a readout layer (trained in a supervised manner) to extract the liquid states and infer the class of the input patterns. The recurrent connectivity and the nonlinear neuronal dynamics enable the liquid to generate high dimensional spike patterns (liquid states) for varied inputs. The information loss is incurred by using fixed input to liquid synaptic connections.

The authors propose SpiLinC that is composed of an ensemble of multiple liquids operating in parallel. They used a "divide and learn" strategy for SpiLinC, where each liquid is trained on a unique segment of the input. SpiLinC incorporates the principle of ensemble learning to recognize an input pattern by training the constituent liquids to extract low-level characteristic features. It rely on the unique nonlinear representations produced by a liquid with fixed recurrent connections to perform recognition. unsupervised training of the liquid to readout connections using STDP would increase the network complexity by necessitating larger number of readout neurons with lateral inhibition.

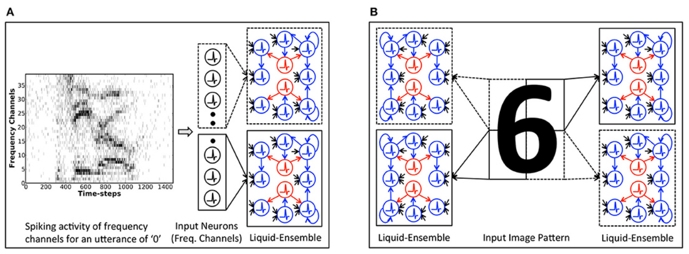


Fig. 3. Illustration of the Spiking Liquid-Ensemble Computing (SpiLinC) architecture composed of a distributed arrangement of multiple liquids operating in parallel. (A) Two-liquid SpiLinC, where each liquid constituting the ensemble is trained with different frequency channels of a speech signal (utterance of "0" in this example). (B) Four-liquid SpiLinC, where the individual liquids are trained with separate partitions of an image pattern.

B. Preparing More Effective Liquid State Machines Using Hebbian Learning (Norton and Ventura, 2006)

An LSM that uses STDP synapses is a Hebbian Liquid State Machine (HLSM). According to this research paper, the effectiveness of an LSM is a function of two qualities: the approximation and the separation of the LSM. Approximation refers to the reading function's ability to classify the state vectors acquired from the liquid. Separation refers to "the amount of separation between trajectories of internal states of the system that are caused by two different input streams"; or in other words, the ability of the liquid to produce discernibly different patterns when given different classes of inputs.In order to measure the separation of a liquid we use the following definition :

$$Sep(\psi, O) = \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{\|C_m(O_i) - C_m(O_j)\|_2}{N^2} \quad (2)$$

where ψ is a neural microcircuit (or liquid), O is a set of state vectors, and N is the total number of output classes represented by o. O is divided into N subsets each of which contains all elements of O belonging to a common output class. The center of mass, C_m for each subset is calculated as follows:

$$C_m(O_i) = \frac{\sum_{o_j \in O} O_j}{|O_i|} \quad (3)$$

There are certain negative behaviors common in LSMs that can significantly decrease their separation. These behaviors are termed pathological synchrony and overstratification. Pathological synchrony occurs when most of the neurons in the liquid get caught in infinite positive feedback loops with respect to their firing. These infinite loops continuously influence the state of the liquid overriding the flow of information from the input. In extreme cases the entire liquid can begin firing continuously in synchrony. Such liquids have low separation because of the loss of pattern associated with such crowded spiking. The opposite extreme is over-stratification—when groups of neurons do not propagate a series of spikes induced by an input spike long enough. In these cases, input spikes do not influence each other within the liquid, thus resulting in a loss of temporal coherency that can be represented by the liquid.

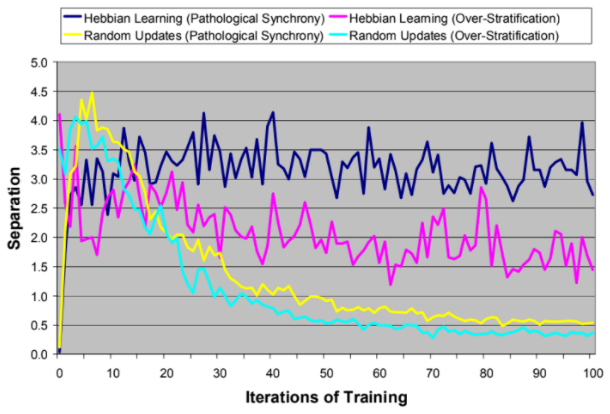


Fig. 4. Separation values for four experiments given completely random input. Separation values are the mean reported by ten trials each with a different initial liquid. The Hebbian learning trials don't show a significant change in separation while the random weight update trials show a steady drop in separation after only ten iterations.

III. Project Idea

The idea that I wanted to try was to investigate the performance of LSM by converting the image from spatial domain to frequency domain. LSM has performed really well on speech datasets like spoken digits where the model gave a test accuracy of 99%. This shows that LSM can effectively learn the temporal patterns in the speech dataset. But the typical image datasets only contain spatial information and not temporal information. Thus, I wanted to try out converting the images to frequency domain using Fourier transform during preprocessing stage. By this, we can expose the image features that are not visible in spatial domain, eg. periodic interferences.

The Fourier Transform is an important image processing tool which is used to decompose an image into its sine and cosine components. Thus, the output of fourier transform is a complex function. The magnitude tells ”how much” of a certain frequency component is present and the phase tells ”where” the frequency component is in the image. Thus, the phase represents the shape in the image.

IV. Dataset

The main dataset used in this project is the CIFAR-10 dataset. It is a collection of images that are commonly used to train computer vision algorithms. It is one of the most widely used datasets for machine learning research. It consists of 60000 32x32 RGB images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The classes are completely mutually exclusive.

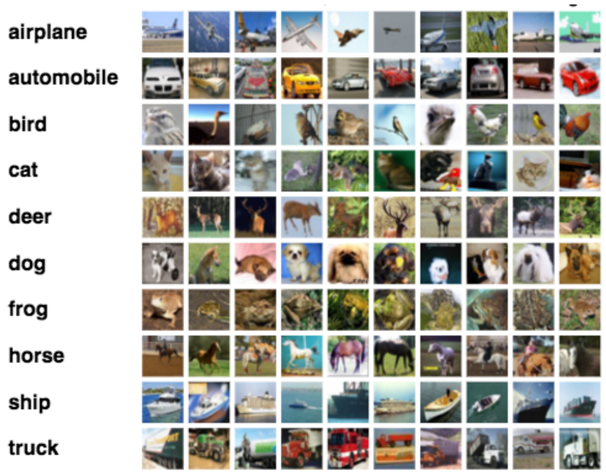


Fig. 5. The above figure shows the classes in the CIFAR-10 dataset and 10 random samples from each

A. Why CIFAR-10 ?

We can distinguish an object from another by visualizing the shape of that object. Since, we are converting the image dataset into frequency domain, we want the image dataset to be scale invariant. Moreover, we want the model to generalize and make it more robust by using various data augmentation techniques like random rotation, random crop, etc. We can't do that with MNIST because if you apply 180 degree rotation to 6 then it becomes 9 and this belongs to a different class. Thus, the model might learn insignificant hidden features and perform poorly. Moreover, MNIST consists of only grey-scale image while CIFAR-10 has 3 channel images. Thus, classification of CIFAR-10 is a more difficult task.

From Fig.6, we can clearly notice the difference in the Phase of the Fourier transform of the images. Thus, the model will be able to capture the shape differences for CIFAR-10 dataset. The two cars have different sizes and orientations and thus have different representation in frequency domain. Thus, our model might be able to generalize by finding out the underlying hidden features and build a robust classifier.

B. Other datasets explored

1) *MNIST:* MNIST is a dataset of handwritten digits of size 28x28 and are gray-scaled. It contains contains 60,000 training images and 10,000 testing images. Currently, LSM is widely tested on this dataset.

2) *N-MNIST & N-Caltech101:* The Neuromorphic-MNIST (N-MNIST) dataset is a spiking version of the original frame-based MNIST dataset. It consists of the same 60000 training and 10000 testing samples as the original MNIST dataset, and is captured at the same visual scale as the original MNIST dataset (28x28 pixels). The Neuromorphic-Caltech101 (N-Caltech101) dataset is a spiking version of the original frame-based Caltech101 dataset (Orchard et al., 2015). These dataset

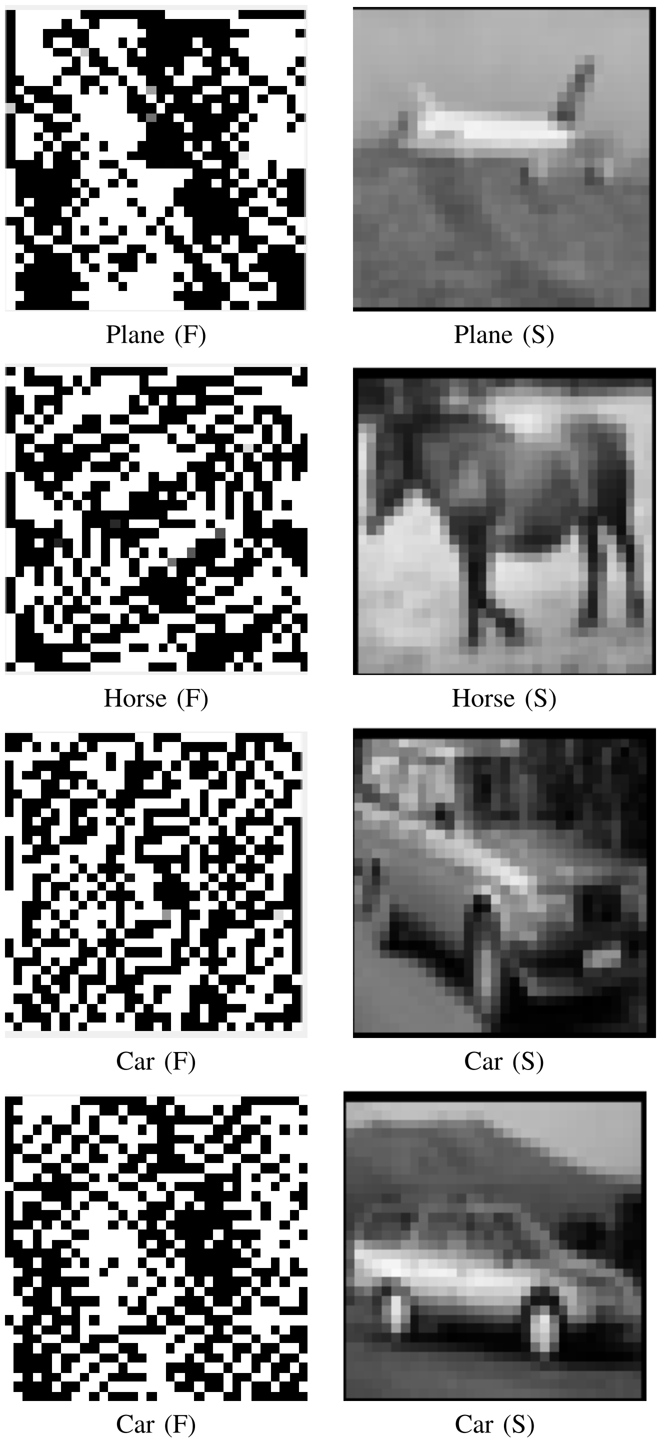


Fig. 6. Left side images denote the Phase of the Fourier transform of the image. The right side images are the corresponding spatial domain. (Note : In the captions, F stands for frequency domain and S for spatial domain)

was captured by mounting the ATIS sensor on a motorized pan-tilt unit and having the sensor move while it views MNIST or Caltech101 examples on an LCD monitor.(Note : A saccade is a quick, simultaneous movement of both eyes between two or more phases of fixation in the same direction. In contrast, in smooth pursuit movements, the eyes move smoothly instead of in jumps.)

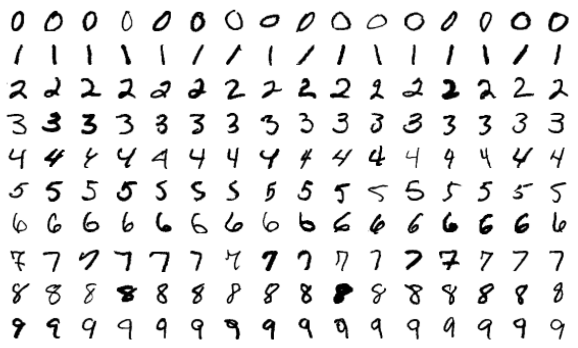


Fig. 7. The above figure shows the classes in the MNIST dataset and 16 random samples from each

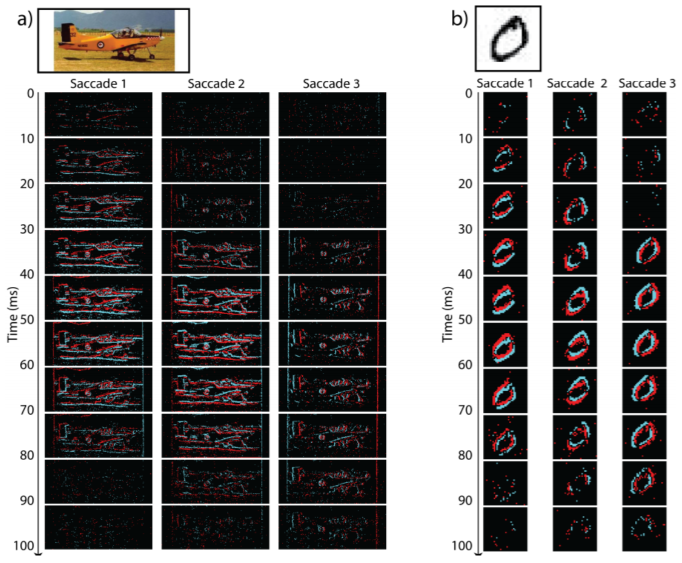


Fig. 8. The above figure shows one example recording from each of the N-Caltech101 (left) and N-MNIST (right) datasets. The original images are shown at the top, with neuromorphic recordings shown below. Each of the neuromorphic subimages contains 10 ms of events. In each case the most events are present near the middle of a saccade when the sensor is moving fastest.

3) CIFAR10-DVS: A Dynamic Vision Sensor (DVS) camera with the spatial resolution of 128×128 was used to convert 10,000 frame-based images into 10,000 event streams (Hongmin Li, 2017).

V. Data Preprocessing

In their paper (Lee et al., 2020), for the color image datasets like CIFAR-10, the authors have used the pre-processing technique of horizontal flip before generating input spikes. These input pixels are normalized to represent zero mean and unit standard deviation. Thereafter, they scaled the pixel intensities to bound them in the range [-1,1] to represent the whole spectrum of input pixel representations. The normalized pixel intensities are converted to Poisson-distributed spike events such that the generated input signals are bipolar spikes.

A. Input spike generation

One approach for representing an CIFAR-10 image using spike trains is to create a set of 32x32x3 = 3072 spike train sequences where each sequence has a firing rate that is proportional to the pixel value. Pixels with small values will be assigned a small firing rate and generate mostly 0-values in their associated spike train. Pixel values that are moderate in magnitude, such as 126 = 7Eh, will be assigned a moderate firing rate and generate roughly 50% spiking 1 values and 50% non-spiking 0 values. Pixel values that are large, such as 255 = FFh will be assigned a large firing rate and generate spike trains that are mostly 1 values.

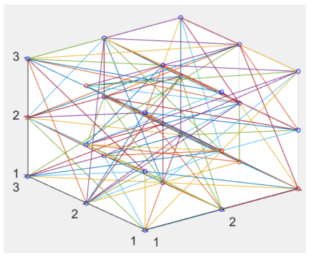
Since the resulting spike trains have a random component, we need to create very long spike trains for each pixel. I chose the the interval of duration **dt** = 0.1ms and number of bins (nBins) = 1000. Thus, for 1 sample image, spike train size would be 3072x1000.

Code for spike train generaion:

```
out_fs = 10000
nBins = 1000
dt = 1/out_fs;
nTrials = numel(dataset.data(1,:));
for i = 1: numel(dataset.data(:,1))
    spikeMat = rand( nTrials , nBins );
    for j = 1:numel(dataset.data(1,:))
    fr = double(dataset.data(i,:));
    spikeMat(j,:) =  spikeMat(j,:) < fr*dt;
    end
    spikeMat =sparse(logical(spikeMat));
end
```

The above code gives us spike train for CIFAR-10 dataset which is in spatial domain. I created another spike train in which Fast fourier transform (FFT) was first applied on all images of the dataset and then the above process was followed again.

VI. Results and Discussions


Reservoir size = 3x3x3

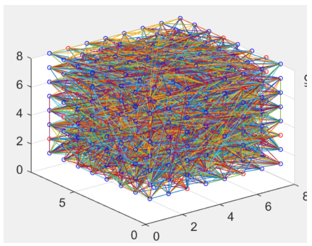

Reservoir size = 8x8x8

Fig. 9. Reservoir connections

I set the training/validation split to 8000/2000. The input was poisson spike trains and the readout layer is a linear classifier.

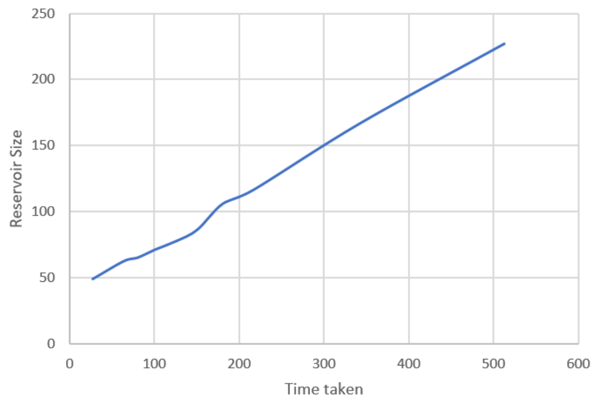


Fig. 10. Variation of time taken (in seconds) for computations for Spoken digit dataset with reservoir size

Reservoir Size	Accuracy CIFAR-10	Accuracy Fourier CIFAR-10	Time taken
5x5x5	26%	20%	2hrs
6x6x6	38%	27%	2.5hrs
8x8x8	48%	36%	4hrs

From the table, we can clearly conclude that validation accuracy increases on increasing the reservoir size. The accuracy achieved for reservoir size of 8x8x8 is reasonable enough when compared to the benchmark as high accuracy is achieved for CNN architectures.

Figure 10 shows variation of time taken (in seconds) for computations with reservoir size. This experiment was performed for spoken digits dataset. It clearly shows that time increases linearly with increase in reservoir size.

Limitations of LSM :

- LSM act as a black box and thus don't explain how the brain functions
- There is no guaranteed way to figure out how or what computations are being performed

VII. Conclusion & Future work

LSM have various advantages over artificial neural networks. Circuits dont need to be hard coded for a specific task and continuous time inputs are handled naturally. They do have limitations as they act as a black box. It seems that standard CIFAR-10 spike train gives better accuracy than the fourier transformed CIFAR-10 spike train. But more experiments need to be done by changing the hyper-parameters of spike train generation and need to analyze the raster plots. Also, more experiments need to be performed by varying the size of the reservoir. GPU acceleration can also be utilized to speed up the computations.

References

Ji Xiangyang Li Guoqi Shi Luping Hongmin Li, Liu Hanchao. Cifar10-dvs: An event-stream dataset for object classification. *Frontiers in Neuroscience*, 11:309, 2017. ISSN 1662-453X. doi: 10.3389/fnins.2017.00309.

Chankyu Lee, Sarwar Syed Shakib, Panda Priyadarshini, Srinivasan Gopalakrishnan, and Roy Kaushik. Enabling spike-based backpropagation for training deep neural network architectures. *Frontiers in Neuroscience*, 14:119, 2020. ISSN 1662-453X. doi: 10.3389/fnins.2020.00119.

Yu Liu, Sai Sourabh Yenamachintala, and Peng Li. Energy-efficient fpga spiking neural accelerators with supervised and unsupervised spike-timing-dependent-plasticity. *ACM Journal on Emerging Technologies in Computing Systems*, 15(3):1–19, 2019. doi: 10.1145/3313866.

W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14:2531–2560, 2002. ISSN 11. doi: 10.1162/089976602760407955.

D. Norton and D. Ventura. Preparing more effective liquid state machines using hebbian learning. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 4243–4248, 2006. doi: 10.1109/IJCNN.2006.246996.

Garrick Orchard, Ajinkya Jayawant, Gregory K. Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in Neuroscience*, 9:437, 2015. ISSN 1662-453X. doi: 10.3389/fnins.2015.00437.

M. T. Sharbati, Y. Du, J. Torres, N. D. Ardolino, M. Yun, and F. Xiong. Low-power, electrochemically tunable graphene synapses for neuromorphic computing. *Communication*, 2018. doi: https://doi.org/10.1002/adma.201802353.

Gopalakrishnan Srinivasan, Priyadarshini Panda, and Kaushik Roy. Spiking liquid-ensemble computing for unsupervised speech and image recognition. *Frontiers in Neuroscience*, 12:524, 2018. ISSN 1662-453X. doi: 10.3389/fnins.2018.00524.

Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose. Recent advances in physical reservoir computing: A review. *Neural Networks*, 115:100 – 123, 2019. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2019.03.005.

Julie Wall and Cornelius Glackin. Spiking neural network connectivity and its potential for temporal sensory processing and variable binding. *Frontiers in Computational Neuroscience*, 7:182, 2013. ISSN 1662-5188. doi: https://doi.org/10.3389/fncom.2013.00182.