Classifying Neuromorphic Datasets with Tempotron and Spike Timing Dependent Plasticity

Laxmi R. Iyer
Machine Intellection
Institute of Infocomms Research
Singapore
Email: laxmi_r_iyer@i2r.a-star.edu.sg

Yansong Chua
Huawei Technologies
Shenzhen, China

***Abstract*—Although the spike rate of a neuron codes useful information, there is a lot of evidence that information is contained in the precise timing of spikes.**

Static images have long been used as benchmarks for ANNs. However, in the neuromorphic community novel benchmarks have developed that have both spatial and temporal information. We require capable SNN algorithms that classify temporal datasets.

There has been a lot of research in training SNNs. Trained ANNs have been converted to SNNs. SNNs have been trained using variants of backpropagation. However, less research has been done on local learning rules, and biologically plausible methods such as spike timing dependent plasticity (STDP) in training SNNs.

In this paper, we present a biologically plausible algorithm that utilizes STDP and tempotron learning rule to classify temporal datasets. We have used DvsGesture to test our algorithm.

***Index Terms*—spiking neural networks, spike timing dependent plasticity, tempotron, self organized feature map, DvsGesture**

I. Introduction

The computational mechanisms of the brain have been a challenge for researchers for the past several decades. Early researchers developed artificial neural networks (ANN), and later, the more biological spiking neural networks were developed. The main advantage of spiking neural networks over their predecessors is the ability to spike. Although the spike rate of a neuron codes useful information, there is a lot of evidence that information is contained in the precise timing of spikes. Primates are able to classify objects around 100-150 ms after the presentation of the stimulus. Given the amount of processing that is necessary for the task, it is not possible to use a time averaged spike code. [1]–[3] Specialized subsystems such as the electrosensory systems of electric fish [4] and the auditory system of barn owls [5], [6] require precise timings of spikes. It has been shown that spike timing is important for the vestibular system [7] and somatosensory system [8]. According to neuroprosthetic studies, precise spike timings are also important in generating smooth movement [9]. From the above studies, we note that a rate code mechanism alone is not adequate in explaining neural coding in the brain. It is therefore important to develop neuromorphic algorithms that account for the temporal coding of neurons.

Images have long been used as benchmarks for artificial neural network algorithms - MNIST was used to test early neural networks, now ImageNet and CIFAR-10 are examples of current day benchmarks. However, these datasets contain spatial data that do not vary with time. As a result, precise spike timings are not required to classify them. Therefore, in the field of neuromorphic computing, new benchmarks have been created, for e.g. N-MNIST [12], N-Caltech101 [12], MNIST-DVS [13] and DvsGesture [36]. Many of these are recorded using the DVS camera [10] and have the time dimension. However, as we argue in [14], many of these datasets are recorded from static images, and therefore do not contain useful information in time (see [14] for more details). In this paper, we use the term 'temporal dataset' frequently. By 'temporal dataset' we mean that patterns in the dataset vary in time, and the manner in which they vary in time is essential information that is required to classify the dataset. DvsGesture is a dataset containing recordings of hand movements, and so it has information that varies in time, therefore it is a temporal dataset.

Spiking neural networks, despite their advantages, have not reached the same performance as their ANN counterparts. In order to improve performance, several approaches have been taken to train SNNs in the recent years. The first approach for training SNNs is to train ANNs, and convert the trained ANNs to SNNs. The trained weights will then be used for inference. The obvious advantage of this is that well-established deep learning techniques can be applied for training before conversion. There are some problems, however. For example, ANNs have negative activations but SNNs fire only spikes. Applying maxpooling in SNNs is not as straightforward as ANNs as maxpooling is a non-linear operation and cannot be applied to spikes. For more information on the issues of converting SNNs to ANNs, see e.g. [15]. Due to these issues, a second approach is to constrain ANNs to have properties similar to SNNs before conversion. Traditional ANN methods are applied to the constrained networks before conversion to SNN. However, converting ANNs to SNNs is evidently not able to take advantage of the temporal properties that are unique to SNN spikes.

This research is supported by Programmatic grant no. A1687b0033 from the Singapore government's Research, Innovation and Enterprise 2020 plan (Advanced Manufacturing and Engineering domain)

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

Another more recent approach of training SNNs is to use variants of backpropagation that are suitable for SNNs. These methods are able to take advantage of the temporal dynamics of spiking neural networks. For example, [16] bin spike trains, and perform gradient descent on the histogram bins. [17] describe a spatiotemporal backpropagation rule that separates spatial input signals that come from other neurons, and temporal dynamics that arise from the neuron itself. [18] has a backpropagation rule that takes into account temporal dependencies in a spike train for credit assignment. In [19] the backpropagation rule updates a rate coded error signal on a macro scale and also has updates on a shorter micro time scale that captures individual spikes. In general, however, backpropagation requires global signals in order to propagate errors throughout the network.

Finally, an approach that is less established is the use of local and biologically plausible training rules. For practical applications, these provide a very hardware-efficient solution to training SNNs. In neuroscience, biologically plausible training of SNNs is of great interest. However, research on STDP methods is relatively new, and results on benchmark datasets have not yet matched other SNN training methods. There are a few researchers that have explored STDP. The group by Masquelier and Thorpe [20], [21] and Panda et. al. [24]–[26] have explored convolutional neural networks (CNNs) in STDP and get very good results on MNIST and other benchmarks used for ANNs. However, their work has been tested on and optimized for static datasets such as MNIST, Caltech-101, etc. Thiele et. al [27], [28] have goals similar to ours, and have produced a CNN with STDP, which achieves very good results on N-MNIST. However, as we have noted in [14], as N-MNIST has very little information in the temporal domain, and does not require an SNN to classify it, and ANNs can do just as well (see [14], Section 3). We, on the other hand, have used DvsGesture, produced from dynamic hand movements to test the dataset.

In this paper, we present the first STDP network that has been tested on a truly temporal dataset. The network described in this paper is an extension of the algorithms described in [29]–[31]. [29] is an interesting implementation of a spiking neural network with STDP, that performs clustering of the input space. [14], [30] has extended this to classify N-MNIST. [31] builds upon [29] by adding the functionality of a self organized feature map (SOM). The network described in this paper builds on earlier works to create an STDP network that is capable of classifying temporal datasets.

Earlier, [32] used a combination of the self organized feature map (SOM) and tempotron [33] to classify an audio dataset, TIDIGITs [32]. However, their SOM was an ANN, and the incoming spikes to the SOM were very regular. Our network is similar to theirs - however, our incoming spikes are from the DvsGesture dataset, and we use a spiking SOM. While they classify audio, DvsGesture is consist of recordings from the DVS camera.

The next section details the methods, followed by experiments and results. The final section is on discussion and conclusion.

II. Methods

A. Network Architecture

The network consists of input layer, excitatory layer and inhibitory layer. The input layer consists of N_{inp} neurons each of which corresponds to a pixel of the input image. The excitatory layer and inhibitory layer consist of N_e neurons. Input neurons activate neurons in the excitatory layer. Each excitatory neuron is connected to an inhibitory neuron with a fixed weight. When an excitatory neuron activates the corresponding inhibitory neuron, the inhibitory neuron inhibits all neurons in the excitatory layer except for the neuron that activated it, with varying degrees of inhibition. Therefore, the functionality of a winner take all (WTA) network is created.

Instead of inhibiting all neurons with a fixed rate as in [29], the level of inhibition is increased in proportion to the square root of the Euclidean distance from the firing neuron, as in the SOM learning algorithm [31]. This is controlled by two parameters, c_{inhib} and c_{max}. The level of inhibition equals c_{inhib} multiplied by Euclidean distance, or c_{max} whichever is less. c_{max} is the maximum level of inhibition. Therefore, when an excitatory neuron fires, neurons that are close by in distance may have a chance to fire, while neurons further away will not. For more information on the inhibition, refer to [31].

Weights between the input neurons and excitatory neurons are plastic. When an excitatory neuron spikes, the input-excitatory weights are learned using STDP. The weights between excitatory and inhibitory layers are fixed.

Threshold Adaptation: If a spike occurs in the excitatory neuron, the threshold is increased so it is harder to spike. The effect of threshold adaptation is that learning is not dominated by a single neuron, but is distributed across the neurons.

Further details of the basic network structure and functionality are described in [29], [30]. Details on SOM functionality are given in [31].

B. Temporal Datasets

In the above network, a neuron fires a spike in response to an input, and due to STDP, learns this input. Due to graded inhibition, several neurons may fire, but learn the same input pattern to varying degrees. This network [29] and its SOM variant [31] have been used to successfully classify MNIST and N-MNIST datasets.

For a pattern that temporally varies over the course of the pattern presentation, such learning is not sufficient. Several independent neurons must spike over the period of pattern presentation during which temporal subpatterns of the main pattern should be learned. Such a sequence of neurons should be used to classify the data.

We enhance the networks described above with this capability by making the following changes.

1) *Short Time constants* - The membrane time constant, τ_M and the synaptic STDP time constant, τ_{STDP} are short, and around $\frac{1}{10}th$ the presentation time. With a short τ_M, spiking occurs regularly over short time periods. With a

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

small τ_{STDP}, learning occurs only when pre and post synaptic spikes occur within a short interval.

2) *Reset of parameters* - At every *k ms* where k is $\frac{1}{10}th$ the presentation time, all voltage traces, all currents and current traces, and synaptic traces are reset. The parameters are reset to ensure that after the *k ms* time window, learning is completely independent of what is learnt in the previous time windows.
3) *Online Learning* - In the previous systems, during pattern presentation, if no spikes arrived, the current (in [14], [30]) or spike rate (in [29]) was increased and the pattern was presented again, to have atleast n spikes. However, in this system, a sequence of spikes from different neurons should occur at regular intervals during pattern presentation. Therefore, the maximum excitatory presynaptic current (EPSC) is kept constant so learning occurs online.
4) *Classification of sequences* - The sequence of output spikes generated by the STDP system are further classified using the tempotron learning rule. The tempotron learning rule is a biologically plausible method for classifying sequences of spikes, and can be used to classify the spikes generated by the STDP system.

Collectively these measures ensure neurons spike at regular intervals throughout the pattern presentation, independently learning temporal subpatterns of the overall input pattern. This is reflected in the learning of weights (Fig. 1). The sequence of neurons that spike are then classified by the tempotron learning rule.

C. DvsGesture

Several neuromorphic benchmarks today are datasets recorded by moving a DVS camera over static images. We argue in [14] that although the temporal dimension exists in these datasets, they do not really have discriminatory features in the temporal domain. DvsGesture is created by recording dynamic arm movements using a DVS camera. Since the arm movements are dynamic, we expect DvsGesture to contain discriminatory features in the time domain, which are necessary information to classify the dataset. We therefore feel that DvsGesture is a suitable dataset to test our algorithm.

DvsGesture is comprised of 1,342 patterns. Arm movements were recorded from 29 subjects who stood against a stationary background. There are 11 arm and hand movements each performed with 3 illumination conditions. These gestures were recorded using the DVS128 ([13]) camera. 11 classes correspond to gestures such as *hand waving*, *arm rotations clockwise*, *arm rotations counter-clockwise* and *clapping*. The 11^{th} class, *Other* consists of a gesture invented by the subject. For ease of classification, we took out the *Other* class.

III. Results

A series of experiments were conducted to test the system. Firstly EPSC value, A_{xe} was adjusted to ensure that most patterns elicited a sequence of output spikes. In the series of experiments described in this section, two parameters were varied, and the system was explored:

1) Number of output neurons, N_e - if the number of output neurons is too small, patterns are not adequately learned. If the number of output neurons is too large, overfitting occurs as there are too many parameters to be trained.
2) Inhibition parameter, c_{inhib} - this is the parameter that controls the neighborhood size in the SOM. If inhibition is too low, the number of neurons that learn independently is too low. Therefore, not enough learning takes place. If inhibition is too high, neurons learn more independently, and overfitting may occur. c_{inhib} is expressed in nA in all results, and the inhibitory post synaptic current (IPSC) is equal to $c_{inhib}\times$ *square root of Euclidean distance from the first spiking neuron.*

A. Adjusting learning and threshold adaptation rate for Network size

Initially, we noted that same amount of A_{xe} current that generates a sequence of output spikes on a larger network (i.e. one with larger number of output neurons) is unable to do so when the network is smaller. This is because when learning is distributed across neurons, in networks with smaller number of output neurons, individual neurons fire more frequently than on networks with larger output neurons. When neurons fire frequently, threshold increases faster (see Section II-A, Threshold Adaptation), and can increase to an extent that no neuron fires. Therefore, if networks of different sizes (i.e. number of output neurons) are to be compared, the threshold adaptation amount should be adjusted for network size. Since learning rate is complementary to threshold adaptation rate (see [14], section 5) learning rate should also be adjusted accordingly. In this section we describe how this is done.

The total amount of threshold increase in the network depends on the number of spikes across all neurons throughout training period (we denote it as N_s) and is equal to:

$$N_s \Delta x \tag{1}$$

where Δx is the amount of threshold increase per spike, in a neuron, or the threshold adaptation rate. If we suppose that all neurons have an equal probability of spiking (this assumption is true if a few neurons do not dominate learning in the network), then the total amount of threshold increase in a particular neuron throughout training can be approximated to

$$\frac{N_s \Delta x}{N_e} \tag{2}$$

where N_e is the number of neurons in the network.

The largest network size used for comparisons is denoted by N_e^*. In the experiments in this paper, we set N_e^* as 900. We set A_{xe} such that a sequence of output spikes are generated for a network of N_e^* neurons. Let the threshold adaptation rate in a network of size N_e^* equal to Δx^*. We adjust Δx for networks of any size, such that the amount of total threshold increase

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

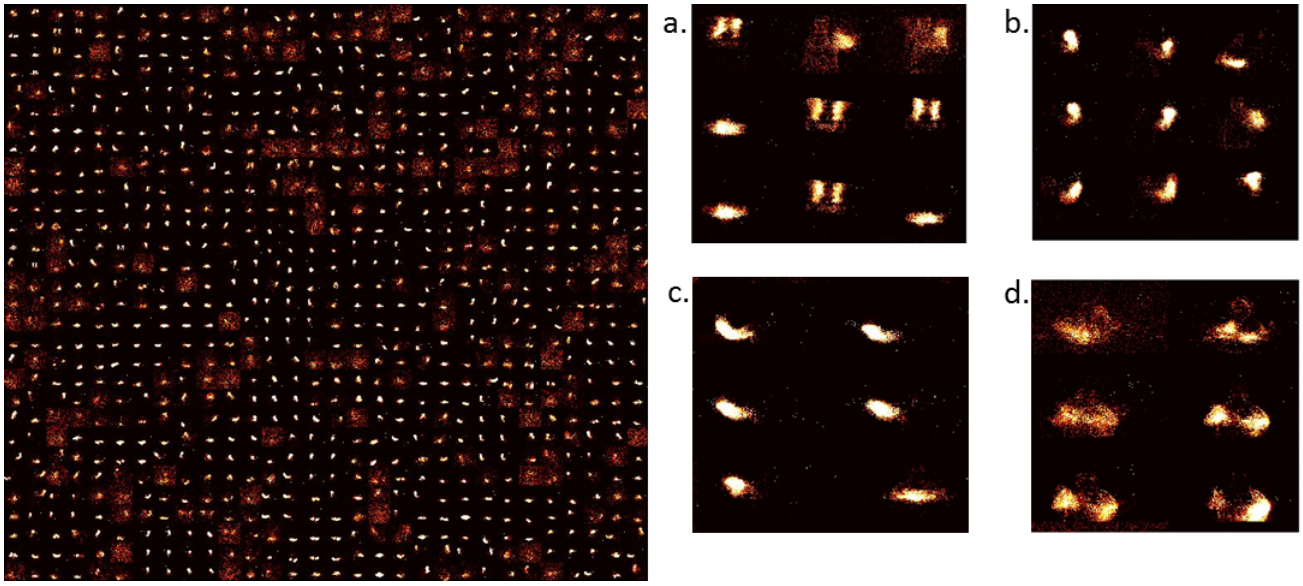

Fig. 1. Input-excitatory weights of a 900 output neuron network after training - *Left:* weights from input to each excitatory neuron is arranged as 128×128 matrix to visualize the input learnt. These indiviual neuron weights are arranged on a 30×30 grid. *Right* (a)-(d): Zoomed snapshots of the image on the left. As can be seen the weights learn temporal snapshots of different actions. These snapshots can be used as raw material for producing actions from different classes. (a.) Some images are a part of *clapping* and others, *arm rolling*, and others, *left hand wave* or *left hand clockwise or counterclockwise* movements, (b.) Can be part of *left-hand waving* or *left hand clockwise or counterclockwise* movements, (c.) can be a part of *right-hand waving* or *right hand clockwise* or *counter clockwise* movements, and (d.) can be part of *arm rolling* or *air drums*. Similar actions are grouped together in space due to SOM functionality.

TABLE I
TRAINING AND TEST ACCURACY FOR NETWORKS OF DIFFERENT SIZES

Number of output neurons	Inhibition (nA)	Training Accuracy	Testing Accuracy
49	10	48.50%	45.38 %
100	10	60.11%	46.82%
400	10	81.01%	52.16%
900	10	86.48%	48.05%

TABLE II
TRAINING AND TEST ACCURACY FOR NETWORKS WITH LESS INHIBITION

Number of output neurons	Inhibition (nA)	Training Accuracy	Testing Accuracy
400	2.5	81.97%	47.84%
400	5	81.56%	51.13%
400	10	81.01%	52.16%

(2) is equivalent to that of the system with N_e^* neurons. This is as follows:

$$\frac{N_s \Delta x^*}{N_e^*} = \frac{N_s \Delta x}{N_e} \tag{3}$$

where N_e is the network size of any network. Readjusting this equation, we get:

$$\Delta x = \Delta x^* \frac{N_e}{N_e^*} \tag{4}$$

On smaller networks Δx is set to the amount calculated in equation 4. Since learning rate and spike frequency adaptation are complementary (see [14] section 5), learning rate is also adjusted in the same manner:

$$\eta = \eta^* \frac{N_e}{N_e^*} \tag{5}$$

where η^* is the learning rate for a network of size N_e^*. In this manner, networks of different sizes can be compared.

In the first experiment, we ran the network for sizes 900, 400, 100, and 49. The inhibition level was set to 10. A sample of the sequence of output neurons firing in response to the input can be seen in Fig. 2. As can be seen, the output spikes are very similar to the input sequence. Both training and test accuracy for each of the networks is shown in table I. As we have seen a 400 neuron network gives the best results in this section. The test accuracy is 52.16%.

B. Training Networks with less Inhibition

Larger networks with more inhibition may lead to more overfitting, since more neurons learn independently of one another. Less inhibition means that neighborhood neurons learn more collectively leading to less number of independent parameters. Therefore, smaller networks were trained with less inhibition. The network size was fixed at 400, and inhibition levels were 10, 5 and 2.5. The results are shown in table II. As can be seen in the results, there is no improvement in results with less inhibition.

In all the results we have seen so far, there is a large difference between the training accuracy and test accuracies. The reason for this is evident - the dataset has only 732 patterns. Input dimensionality is 128×128 input neurons. Weights are of the size $128 \times 128 \times$ *number of output neurons*.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

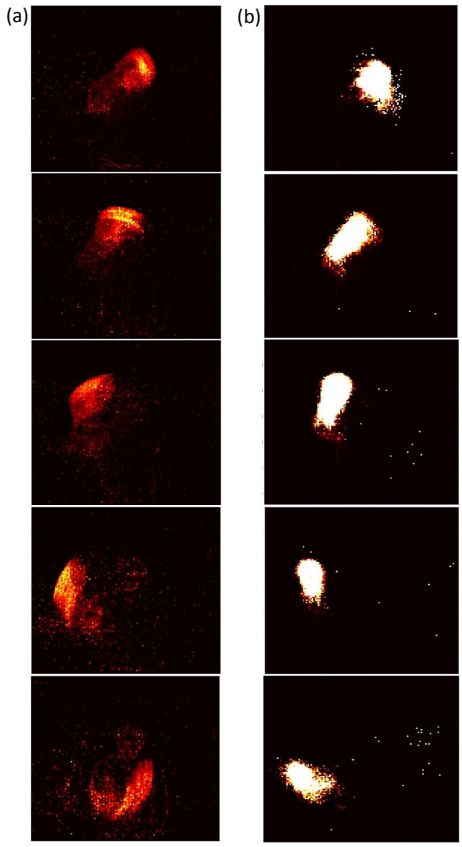

Fig. 2. (a.) Sequence of frames created from input spikes. Input spikes collapsed over every m ms, where m is $\frac{1}{10}th$ the presentation time. The images from top to bottom depict a pattern from the class *right hand clockwise*. (b.) The images from top to bottom show the sequence of output neurons that spike in response to the input. Each output neuron is represented by weights from the input, and rearranged on a 128×128 grid as in fig. 1. As can be seen, the sequence of output neurons that fire on the right, look very similar to the input spikes on the left.

Therefore, the number of parameters to be trained is very high, and the dataset is not adequate to train it. In the next section, we will examine ways to overcome overfitting.

C. Methods to overcome Overfitting

Reducing the number of training parameters or increasing the number of patterns in the training set are two methods to reduce overfitting. We will explore methods for doing both in this section.

1) Maxpooling: One of the popular deep learning methods to overcome overfitting is maxpooling. We apply maxpooling to the input patterns, so they are reduced from a dimensionality of 128×128 to that of 64×64 and 32×32 respectively. In the DvsGesture dataset, patterns consist of AER events rather than intensity values at pixels. We, therefore do maxpooling of the events. If an event exists in any of the pixels in the $n \times n$ patch of the input pattern at time t, an event is set at the corresponding pixel of the maxpooling output at time t.

Maxpooling was applied using both $n = 2$, i.e. 2×2 as well as $n = 4$, i.e. 4×4 filters (see Fig. 3). Results obtained from this is given in Tables III and IV. As can be seen, maxpooling

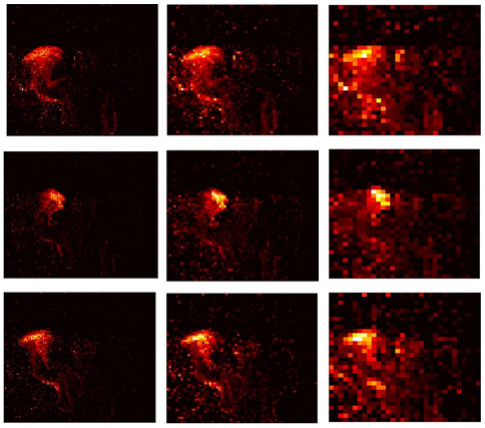

Fig. 3. *Left:* Sequence of frames created from input spikes from a 128×128 input. Input spikes collapsed over every m ms, where m is $\frac{1}{10}th$ the presentation time. The images from top to bottom depict a pattern from the class *right hand wave*. In the two columns on the right, the original 128×128 pattern is maxpooled to create a 64×64 pattern (middle column), and 32×32 pattern (right).

TABLE III
TRAINING AND TEST ACCURACY FOR MAXPOOLING WITH 2×2 FILTERS

Number of output neurons	Inhibition (nA)	Training Accuracy	Testing Accuracy
100	10	60.93%	48.46%
400	10	83.33%	48.25%
900	10	80.19%	39.63%

does not improve the results, with the best results of 48.46% not exceeding the best results in the previous section.

2) Data Augmentation: Another technique we used for overfitting was data augmentation. An important consideration to be taken while producing augmented images with DvsGesture is that the dataset is represented using address-event representations (AER) rather than just intensity values at pixels as in regular images. Therefore, the transformation should be applied to each event rather than pixel value. An event has an (x, y) coordinate along with a time stamp t and polarity p so we denote this event with the tuple (x, y, t, p). Image augmentation is applied only to the training patterns and not the test patterns.

We do mirroring the events along the y-axis and shifting the pixels along x- and y- axes, and have 7 total transformations.

The first transformation is mirroring. In a pattern, the x-coordinate of each event is mirrored along the y-axis, as follows. Suppose x_{max} is the maximum value of the x-coordinate. So x^m, the transformed x-coordinate is:

$$(x, y, t, p) \rightarrow (x^m, y, t, p) | x^m = x_{max} + 1 - x \qquad (6)$$

Applying this transformation to all events in pattern P_1 produces a new pattern P_2 which is a reflection of all events along the y-axis in P_1. Note that for patterns of some classes, the mirrored pattern belongs to a different class. For example if pattern P_1 is *right hand wave* the mirrored pattern will be *left*

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

TABLE IV
TRAINING AND TEST ACCURACY FOR MAXPOOLING WITH 4×4 FILTERS

Number of output neurons	Inhibition (nA)	Training Accuracy	Testing Accuracy
100	10	58.47%	35.93%
400	10	81.56%	47.84%
900	10	88.93%	41.27%

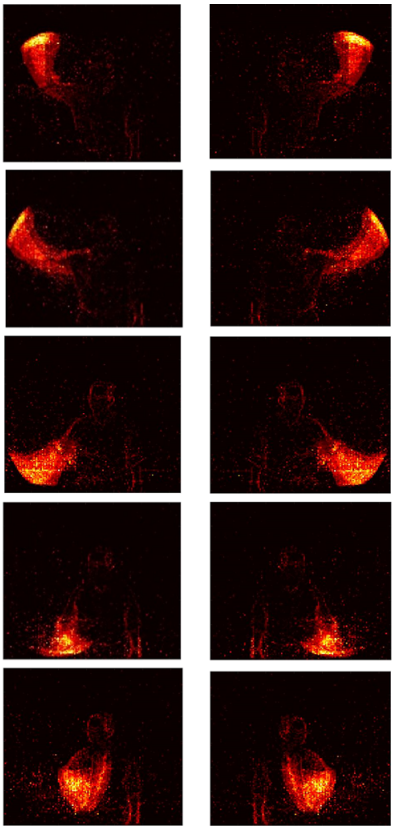

Fig. 4. *Left:* Original pattern: Sequence of frames created from input spikes from a 128×128 input. Input spikes collapsed over every m ms, where m is $\frac{1}{10}th$ the presentation time. The images from top to bottom depict a pattern from the class *right hand clockwise*. *Right:* Frames from the pattern that has been produced as a result of applying the *Mirroring* transformation (Equation 6) to the pattern shown on the left. This pattern is of class *left hand counterclockwise*.

hand wave. If P_1 is *right arm clockwise*, the mirrored pattern P_2 will be *left arm counter clockwise*. This can be seen in Fig. 4. We have applied the transformation in Equation 6 patterns of all classes except *air guitar*. This is because we expect guitar players to predominantly use one side for playing guitar, and do not want to change the characteristics of the original dataset.

The next transformations involve shifting the events by a few pixels in the x- and y-axes. The shifting transformations are applied to the original patterns as well as the patterns produced by the mirroring transformation (Equation 6).

$$(x, y, t, p) \rightarrow (x_1^s, y, t, p) | x_1^s = x + 2 \tag{7}$$

TABLE V
TRAINING AND TEST ACCURACY FOR DATASET WITH AUGMENTED DATA

Number of output neurons	Inhibition (nA)	Training Accuracy	Testing Accuracy
100	10	48.23%	43.33%
400	10	70.37%	56.06%
900	10	83.23%%	60.37%

This involves the shifting of the image to the right by 2 pixels. Similarly, image was shifted to the right by 4 pixels, to the left by 2 and 4 pixels respectively, shifted up by 2 pixels and down by 2 pixels.

So each pattern P_0 will produce 6 new patterns, $P_1 - P_6$, one pattern for each transformation. The class of the shifted pattern will remain the same as the original pattern. Those events whose address after shifting falls outside the 128×128 are removed from the new patterns.

All the transformations described above augmented the size of the dataset from 732 to 5777. Results from running the system with the augmented images are shown in Table V.

As can be seen, data augmentation does increase the test accuracy to 60.37%, a significant improvement over the previous results.

IV. DISCUSSION AND CONCLUSION

In this paper, we present an SNN with STDP learning, SOM and tempotron for temporal datasets and test it with the DvsGesture dataset. We note that the best results with the DvsGesture dataset is 60.37%.

Earlier networks have been tested on DvsGesture, and their results are much better. Maro and Benosman's method [22] has an accuracy of 96.6%, The system proposed by Yang et. al. [23] gets 97.4% accuracy, SLAYER has 93.64% accuracy and Amir et. al. [12] gets 96.49% accuracy. [22] and [23] use advanced computer vision techniques in their networks. Amir et. al [12] and SLAYER [18] use backpropagation with deep neural networks for classification. The network described in [12] has 16 layers, while SLAYER has 8 layers.

However, our system uses the biologically plausible learning mechanisms for classification. As the main purpose of this network is to demonstrate SNN learning on a temporal database, we do not use additional computer vision techniques in this paper. Since CNNs using STDP are just being researched, we first performed this experiment on a single layer feedforward network which is obviously a much simpler network than CNNs with backpropagation described above. Further work will be done on such deep networks to avoid overfitting and for enhanced accuracy.

It should be noted that research in this area is important, but relatively new. SNNs that use STDP have been recently developed (for e.g. [20], [21], [24]–[29]), but they have either been optimized for static datasets such as MNIST and Caltech-101, or neuromorphic datasets derived from static images such as N-MNIST (which we argue in [14] does not have discriminative information in the time domain), but as far as we know none have been tested on temporal datasets.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

The highlights of this paper are:

- SNN with biologically plausible learning rules - STDP, SOM and tempotron.
- Tested on a temporal dataset, DvsGesture. DvsGesture consists of dynamic hand gestures, which need information of temporal sequences in order to classify it.
- Online learning - Unlike [29], the network on which this paper was based, this network learns online from a stream of AER data, without requiring a pattern to be presented again, if it did not elicit spikes.
- Clustering of the dataset (first layer) is completely unsupervised. The supervised tempotron is used only for learning sequences of output spikes. Error signals are not propagated to the first layer to improve the clustering results.

As there is a large difference between the training and test accuracy, we can see that the system has overfitting. Further, when the data is augmented there is an improvement in the results, showing that the small dataset is inadequate to train all the parameters of the network, due to all-to-all connectivity. The issues in this paper that need to be addressed are as follows:

- We need to move towards CNNs, rather than all-to-all connection networks.
- Rather than having artificial time steps after which all parameters are reset, the system should be more event-based [27] - Rather than having artificial clocking mechanisms such as system reset, learning should be triggered by events. In this manner, when there are no events, there is no learning. This is important for energy efficiency.

Examples of CNNs with STDP that have recently been developed are [20], [21], [24]–[28]. More temporal datasets need to be developed. DvsGesture and N-Cars [36] are examples of datasets that are being developed to address the issues with neuromorphic datasets derived from static images.

In the spirit of neuromorphic systems, we need to move closer to biological mechanisms. Research in STDP based mechanisms is still nascent, with results generally not being equal to those with backpropagation or converting SNNs to ANNs. This research is an important step in that direction.

ACKNOWLEDGMENT

The authors would like to thank Jibin Wu for helping with the tempotron.

REFERENCES

[1] H. Kirchner and S. Thorpe, "Ultra-rapid object detection with saccadic eye movements: Visual processing speed revisited," Vision Research vol. 46, pp. 17621776

[2] S. Crouzet and H. Kirchner and S. Thorpe, "Fast saccades toward faces: Face detection in just 100 ms," Journal of Vision, vol. 10, pp. 117

[3] D. Butts and C. Weng and J. Jin and C. -I. Yeh and N. Lesica and J. -M. Alonso and G.B. Stanley, "Temporal precision in the neural code and the timescales of natural vision," Nature vol. 449, pp. 9295, doi:10.1038/nature06105

[4] W. Heiligenberg, "Neural Nets in Electric Fish," Cambridge: MIT Press, 1991

[5] W. Gerstner and R. Kempter and J. van Hemmen and H. Wagner, "Hebbian learning of pulse timing in the barn owl auditory system," In Pulsed Neural Networks, eds. W. Maass and C. Bishop, pp. 351375, MIT Press, 1999.

[6] C. Carr and M. Konishi, "A circuit for detection of interaural time differences in the brain stem of the barn owl," Journal of Neuroscience, vol. 10, pp. 32273246, 1990.

[7] S. G. Sadeghi, M.J. Chacron and M. C. Taylor and K. E. Cullen, "Neural variability, detection thresholds, and information transmission in the vestibular system," Journal of Neurosci., vol. 27, pp. 771781, 2007, doi:10.1523/JNEUROSCI.4690-06.2007

[8] M. Harvey and H. Saal and J.F. Dammann III and S. J. Bensmaia, "Multiplexing stimulus information through rate and temporal codes in primate somatosensory cortex," PLoS Biology 11, 2013 doi:10.1371/journal.pbio.1001558

[9] D. Popovic and T. Sinkjaer, "Control of Movement for the Physically Disabled," London: Springer-Verlag, 2000

[10] G. Orchard and G. Cohen and A. Jayawant and N. Thakor, "Converting Static Image Datasets to Spiking Neuromorphic Datasets Using Saccades," Front. Neurosci., vol.9, no.437, Oct. 2015

[11] T. S.-Gotarredona and B. L.-Barranco, "Poker-DVS and MNIST-DVS. Their History, How They Were Made, and Other Details," Front. Neurosci., vol. 9, no. 481, 2015, doi:10.3389/fnins.2015.00481

[12] A. Amir and B. Taba and D. Berg and T. Melano and J. McKinstry and C. D. Nolfo, "A Low Power, Fully Event-Based Gesture Recognition System," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017

[13] T. S.-Gotarredona and B. L.-Barranco,"A 128×128 1.5% Contrast Sensitivity 0.9% FPN 3 s Latency 4 mW Asynchronous Frame-Free Dynamic Vision Sensor Using Transimpedance Preamplifiers," IEEE Journal of Solid-State Circuits, vol. 48, no. 3, pp. 827-838, 2013

[14] L.R. Iyer and Y. Chua and H. Li, "Is Neuromorphic MNIST neuromorphic? Analyzing the discriminative power of neuromorphic datasets in the time domain," arXiv 2018

[15] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: opportunities and challenges," Front. Neurosci., 2018, doi: 10.3389/fnins.2018.00774.

[16] E. Stromatias and M. Soto and T. Serrano-Gotarredona and B. Linares-Barranco, "An event-driven classifier for spiking neural networks fed with synthetic or dynamic vision sensor data," Front. Neurosci., 11:350, 2017, doi: 10.3389/fnins.2017.00350.

[17] Y. Wu and L. Deng and G. Li and J. Zhu and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," arXiv [Preprint], arXiv:1604.03058, 2017.

[18] S. B. Shreshtha and G. Orchard, "SLAYER: Spike layer reassignment in time," 32nd Conference on Neural Information Processing Systems (NeurIPS), 2018.

[19] Y. Jin and P. Li and W. Zhang, "Hybrid macro/micro level backpropagation for training deep spiking neural networks," arXiv [Preprint], arXiv:1805.07866.

[20] S.R. Kherapidsheh and M. Ganjitabesh and S.J. Thorpe and T. Masquelier, "STDP-based spiking deep convolutional neural networks for object recognition," Neural Networks, vol. 99, 2017, doi:10.1016/j.neunet.2017.12.005

[21] M. Mozafari and M. Ganjitabesh and A. Nowzari-Dalini and S.J. Thorpe and T. Masquelier, "Bio-inspired digit recognition using reward-modulated spike-timing-dependent plasticity in deep convolutional networks," Pattern recognition, vol. 94, pp. 87-95, 2019.

[22] J-. M. Maro and R. Benosman, "Event-based gesture recognition with dynamic background suppression using smartphone computational capabilities", arXiv [preprint] arXiv:1811.07802v2 [cs.CV], 2019

[23] J. Yang and Q. Zhang and B. Ni and L. Li and J. Liu and M. Zhou and Q. Tian, "Modeling point clouds with self-attention and gumbel subset sampling", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019

[24] P. Panda and K. Roy, "Unsupervised regenerative learning of hierarchical features in spiking deep networks for object recognition," 2016 International Joint Conference on Neural Networks, July 2018.

[25] C. Lee and P. Panda and G. Srinivasan and K. Roy, "Training deep spiking convolutional neural networks with STDP-based unsupervised pre-training followed by supervised fine-tuning," Front. Neurosci., vol. 12 no. 435, 2018, doi:10.3389/fnins.2018.00435

[26] C. Lee and G. Srinivasan and P. Panda and K. Roy, "Deep spiking convolutional neural network trained with unsupervised spike-timing-

dependent plasticity," IEEE Transactions on Cognitive and Developmental Systems, vol. 11, no. 3, 2019.
[27] J. C. Thiele and O. Bichler and A. Dupret, "A timescale invariant STDP-based spiking deep network for unsupervised online feature extraction from event-based sensor data," 2018 International Joint Conference on Neural Networks, July 2018.
[28] J. C. Thiele and O. Bichler and A. Dupret, "Event-based, timescale invariant unsupervised online deep learning with STDP," Front. Comp. Neurosci., vol. 12, no. 48, 2018, doi: 10.3389/fncom.2018.00046
[29] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing dependent plasticity," Front. Comp. Neurosci., Aug. 2015
[30] L. R. Iyer and A. Basu, "Unsupervised learning of event-based image recordings using spike-timing-dependent plasticity," 2017 International Joint Conference on Neural Networks (IJCNN), May 2017.
[31] H. Hazan and D. Saunders and D. T. Sanghavi and H. Siegelmann and R. Kozma, "Unsupervised Learning with Self-Organizing Spiking Neural Networks," 2018 International Joint Conference on Neural Networks (IJCNN), July 2018.
[32] Z. Pan and H. Li and J. Wu and Y. Chua, "An event-based cochlear filter temporal encoding scheme for speech signals," 2018 International Joint Conference on Neural Networks (IJCNN), 2018.
[33] R. Gutig and H. Sompolinsky, "The tempotron: a neuron that learns spike-timing based decisions," Nature Neuroscience, vol. 9, no. 3, Mar. 2006
[34] S. Johansson and I. Birznieks, "First spikes in ensembles of human tactile afferents code complex spatial fingertip events," vol. 7, pp. 170177, 2004
[35] F. Gabbiani and J. Midtgaard, "Neural information processing," Encyclopedia of Life Sciences (Nature Publishing Group), pp. 112, 2001
[36] A Sironi and M. Brambilla and N. Bourdis and X. Lagorce and R. Benosman, "HATS: Histograms of Averaged Time Surfaces for Robust Event-based Object Classification," To appear in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018

978-1-7281-6926-2/20/$31.00 ©2020 IEEE