Enabling Homeostasis using Temporal Decay Mechanisms in Spiking CNNs Trained with Unsupervised Spike Timing Dependent Plasticity

Krishna Reddy Kesari*, Priyadarshini Panda+, Gopalakrishnan Srinivasan* and Kaushik Roy*
*School of Electrical and Computer Engineering, Purdue University +Department of Electrical Engineering, Yale University
United States of America
kkesari@purdue.edu

***Abstract*—Convolutional Neural Networks(CNNs) have become the work horse for image classification tasks. This success has driven the exploration of Spike Time Dependent Plasticity (STDP) learning rule applied to the convolutional architecture for complex datasets as opposed to the fully connected architecture. Inhibitory neurons and adaptive threshold are widely adopted methods of inducing homeostasis in fully connected spiking networks to aid the unsupervised learning process. These methods ensure that all neurons have approximately equal firing activity across time and that their receptive fields are different, generally referred to as homeostatic behavior. While the adaptive threshold is straightforward to implement in spiking CNNs, adding inhibitory neurons is not suitable to the convolutional architecture due to its shared weight nature. In this work, we first show that adaptive threshold in isolation is weak in obtaining approximate equal firing activity across activation maps in a spiking CNN. Next, we develop weight and offset decay mechanisms that enable the desired behavior to complement the STDP learning rule and adaptive threshold. We empirically show that these decay mechanisms improve feature learning as compared to baseline STDP in terms of accuracy (up to 1.4%) as well as enhanced homeostatic behavior among activation maps (more than halving the standard deviation). We discuss the complementary behavior of the decay mechanisms as compared to the adaptive threshold in terms of the variance in the activity induced. Finally, we show that when the convolutional features are trained on a subset of classes using STDP with decay mechanisms, the features learned are transferable to the subset of classes that are unseen to the convolutional layers. Thus, the decay mechanisms not only encourage the network to learn better features corresponding to the task being trained for but learn common structure prevalent among the classes while encouraging contribution from all activation maps. We perform experiments and present our findings on the Extended MNIST (EMNIST) dataset.**

***Index Terms*—spiking CNNs, homeostatic behavior, temporal decay mechanisms**

I. Introduction

OVER the last decade, enormous increase in the availability of data and compute power have led to the development of powerful algorithms to perform pattern recognition. The most successful class of algorithms gave rise to the deep learning revolution that has led to the development of numerous supervised [1], [2] and reinforcement [3] learning algorithms . However, two areas that require substantial work are unsupervised learning from an algorithmic perspective and power efficiency from a hardware perspective. Spiking Neural Networks (SNNs), touted as the third generation of neural networks, show potential to address these issues. In contrast to deep learning methods, SNNs consist of neuron models governed by differential equations and learning methods inspired from spiking activity observed in neuroscience experiments [4], [5]. SNNs use spikes at different time instants to transmit information from one neuron to another. In strong association with the brain, time is an inherent characteristic of such networks while information transfer in terms of spikes is widely believed to be the reason for the power efficiency. A commonly used unsupervised learning rule to train SNNs is the Spike Time Dependent Plasticity (STDP), which defines the update of the weight between two neurons, referred to as the pre and post neurons, to be inversely proportional to the time that the post neuron takes to fire relative to the time at which the pre-neuron had last fired.

Akin to multi-layer perceptron inspired architectures in deep learning, initial models of SNNs trained using STDP exhibited a fully connected architecture [6]. The success of the Convolutional Neural Network (CNN) architecture for vision tasks has pushed for the exploration of STDP to spiking CNNs. We mention a select subset of the multiple methods that have emerged to train spiking CNNs using STDP. Layer-wise spiking representation learning approaches have been implemented in multiple flavors [7], [8]. An alternate approach uses a Difference of Gaussians (DoG) filtered input for the convolutional layers trained using STDP [9]. An extension introduces inter and intra map inhibitory layers to remove unfavorable interactions in the Winner Take All (WTA) layer being trained and subsequently removed after training [10].

In all the above works, we note the presence of inhibitory neurons and adaptive threshold as mechanisms to induce homeostasis. Homeostasis encourages a dynamic system to regulate its activity such that it maintains an approximate equal activity across its constituents over a time duration. This is essential in SNNs trained using unsupervised methods

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

to ensure that all the constituent elements of the network play a contributing role. It is relatively easier to define the homeostatic action in a fully connected architecture, wherein inhibitory neurons and adaptive threshold enforce different receptive fields among neurons in a complementary manner. In the event of a certain excitatory neuron firing, inhibitory neurons restrict any other excitatory neuron from firing. On the other hand, adaptive threshold of the neuron that fired is increased so as to enable other neurons to play a contributing role in a different time instant. However, in spiking CNNs trained using unsupervised STDP, incorporating inhibitory neurons is difficult due to their shared weight nature across multiple neurons. The shared weight structure (and common feature learning) is inherent to the success of the CNN architecture and its generalizing capabilities. In order to circumvent this issue, previous works specifically state a restriction on the initialization [9], which maybe necessary in part due to absence of mechanisms to enable a reasonable degree of homeostasis. The method of adding inhibition highly specific to WTA feature representations [10] do not capitalize on the ability to learn shared feature representations provided by CNNs. In a different work, the norm of weights is used for homeostasis, much like regularization used in deep learning, which does not capture the temporal activity inherent to the network [11].

On the other hand, recent work has shown that temporal decay in synapses of fully connected networks aids in learning better features [12]. Drawing inspiration, our approach to employing decay mechanisms is motivated from the observation that the absence of inhibitory neurons leads to a large difference in the activity of the activation maps. This is attributed to the difference in the proportion of positive weights in the weight kernel. As the weight updates according to the STDP rule are performed in the event of a post spike, kernels with smaller positive and negative weights are rarely updated as the frequency of these kernels causing a post spike is low. In addition, the weight updates are small as the time difference between the pre and post spike tend to be longer. Thus, all kernels do not contribute equally in the classification process and the activity tends to be dictated in part by the proportion of positive weights during initialization. In this work, we show that decay mechanisms in spiking CNNs could bring about the necessary homeostatic behavior encouraging all activation maps to have approximately equal activity across time. In addition, we show that decay mechanisms lead to better feature learning in the convolutional architecture using accuracy as the metric. The decay mechanisms are temporally dependent on the kernel weights and the activity produced by the kernel which is reflected in the threshold of neurons in the activation maps. To the best of our knowledge, this is the first work that presents decay mechanisms in spiking CNNs trained with STDP to enhance the learning process and to enable a higher degree of homeostasis in a controllable manner.

Overall, the key contributions of our work are:

1. We present methods to formulate decay mechanisms in synapses and the offset term while training spiking CNNs using unsupervised STDP. We show that these formulations lead to a controllable homeostatic behavior in the absence of inhibitory neurons with each convolutional filter exhibiting comparable activity in their respective activation maps.

2. The same formulations also lead to better feature learning and increased test accuracy. In addition, we also show that modulating the learning rate coupled with decay mechanisms further improves learning.

3. We show that the convolutional features learned by unsupervised STDP are not specific to certain classes. The improved feature learning using the decay mechanisms retains this characteristic as there is an increase in accuracy for the classes being trained as well as for the classes that are not shown during training.

The rest of the paper is organized as follows. In Section 2, we detail the fundamentals of spiking CNNs and the architecture used in this work. This is followed by identifying the parameters of interest, formulation of decay mechanisms and training methodology in Section 3. Section 4 constitutes results and discussion of experiments performed on the full dataset and transfer learning before concluding in Section 5.

II. Fundamentals

A. Spiking Neuron Model

The fundamental unit of an SNN is a spiking neuron. At any time instant, a spiking neuron exhibits one of two states, it either emits a spike (denoted as state 1) or not (denoted as state 0). We use a spiking neuron referred to as a Leaky Integrate and Fire (LIF) neuron which is parameterized by its membrane potential V_{mem}, threshold potential v_{th} and the leak constant τ. The input I_{post}/G to the neuron at a time step is the activity of the neurons it is connected during the previous time step weighted by the plasticity of their connection. The membrane potential of the neuron that receives this input is increased by the sum of the inputs from all the neurons. The membrane potential of the LIF neuron is decayed by an amount dictated by the leak constant τ at every time instant. The neuron equation is given by

$$\tau \frac{V_{mem}}{dt} = -V_{mem} + I_{post}/G \tag{1}$$

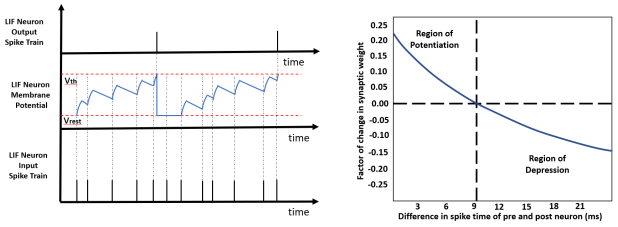
A spiking neuron in which the membrane potential is not decayed is referred to as the Integrate and Fire (IF) neuron. A spiking neuron emits a spike at the time instant its membrane potential reaches the threshold voltage as shown in Figure 1a. Once the neuron emits a spike at a time instant, its membrane potential is reset. Apart from the resetting of the membrane potential, the threshold of the neuron is increased if an adaptive threshold is used.

B. Synapse Model and STDP

The synapses in our spiking CNN are updated according to the unsupervised STDP rule given by Equation 2, wherein the update is proportional to the difference in the time of spiking of the pre and post neuron and scaled by a term that is a function of the present weight [13]. Potentiation (increase) in

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

weights happens up to a certain difference in time dictated by the offset term in Equation 2 exceeding which leads to depression (reduction) in the weight of the synapse as shown in Figure 1b. In the absence of a temporal decay in the synapses, the update is only dictated by Equation 2, where t_{pre} and t_{post} are the time instants of spiking activity of pre and $post$ neurons respectively, η is the learning rate, w_{max} and w_{min} are bounds on the weights with w being the present weight. In the presence of a weight decay, the weights move towards zero based on their temporal activity at every time instant. The presence of temporal decay in the offset term works in a similar manner.



(a) (b)

Fig. 1: a) LIF Neuron and b) Weight update using STDP

$$\delta w = \eta * (\exp(\frac{t_{pre} - t_{post}}{\tau}) - offset) * (w_{max} - w) * (w - w_{min}) \tag{2}$$

Next, we briefly describe the adoption of Equation 2 to the minibatch scenario in CNNs. At each time step, the minibatch leads to spiking of post neurons in the Conv1 layer as the weight kernel is moved over the input. The spiking activity of each post neuron is normalized across the minibatch. Following this, we identify the inputs for which each post neuron had spiked. This leads to a weight update of the corresponding kernel that is proportional to the time difference between spiking of this post neuron and the $k \times k$ pre-neurons according to Equation 2. We follow the implementation in [14] apart from using real valued weights and refer the reader to the same for further details. The key detail that we are interested in is the adaptive threshold of an activation map. As the threshold of the activation map is dictated by its activity, the threshold value at any time instant is indicative of its past activity. All neurons in an activation map are governed by the same adaptive threshold, loosely referred to as the adaptive threshold of the activation map. The threshold is initialized to zero before training and the update of the adaptive threshold is a function of the activity of the map at a time instant is given by

$$\Delta thresh = \beta_{thresh} \times \frac{output\ spike\ count}{output\ map\ size} \tag{3}$$

C. Architecture

The architecture of the spiking CNN used in this work consists a convolutional layer followed by pooling layer which feeds into a softmax layer of dimension equivalent to the number of output classes as shown in Figure 2. The convolutional layer consist of LIF neurons while the pooling layer is composed of Integrate and Fire (IF) neurons. We do not add any hidden layers that are trained using backpropagation as we are interested in understanding the effect of decay on the convolutional feature learning and the softmax directly taps into the these convolution layers. The input to this network is encoded as a Poisson spike train which is fed into the spiking neurons in Conv1 as the convolutional kernels move across the input each time step. The firing rate of the Poisson spike train for each pixel is governed by its intensity. The input, which is either 0 or 1 at a given time instant in multiplied by the corresponding weights of the kernels. The membrane potential of the post neuron in the Conv1 layer to which the input is fed into is increased by this value. The time instant at which the membrane potential of the neuron reaches its threshold voltage, it emits a spike or its state at that specific time step is 1. The output spikes at the convolutional layer Conv1 are accumulated, pooled in AvgPool1 layer and fed to a softmax layer which is trained using traditional backpropagation.

In the transfer learning experiments, the convolutional layers are trained using STDP over a subset of classes, and not all classes. The STDP training hyperparameters are chosen so as to maximize the accuracy of the subset of classes it is being trained over. The learned feature maps are then analyzed for their performance on unseen classes. With respect to EMNIST, we define Task 1 on which the convolutional layers are trained to encompass first 13 alphabets of the English language while the Task 2 refers to the rest of the alphabets.

III. Methodology

The weights of the convolutional layer are learned using unsupervised STDP. All experiments are performed using PyTorch [15]. It is worth noting that kernels initialized using the default PyTorch (Kaiming) initialization perform reasonably well with CNN architecture. This is due to the inherent local structure preserving nature of CNNs that is a well matching inductive bias for vision data. In addition, the initialization bears interesting properties [16]. Incorporating homeostatic behavior with the widely adopted initialization provides the best of both worlds. Thus, we use weights initialized using the above method as our baseline for STDP learning. The STDP learning would in turn serve as a baseline for STDP coupled with weight and offset decay mechanisms.

We perform experiments on the full dataset as well as a subset of the data in order to show that the features learned are the common structure possessed among the dataset. In the complete dataset scenario, both the convolutional layers and the softmax are trained with all classes of the dataset albeit different amounts of data. The transfer learning scenario involves convolutional layers trained on only a subset of classes using unsupervised STDP which feed into two softmax layers, each of which are trained on different subsets of the data. We are keen on understanding the effect of the decay mechanisms on the generalization of convolutional features learned using STDP.

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

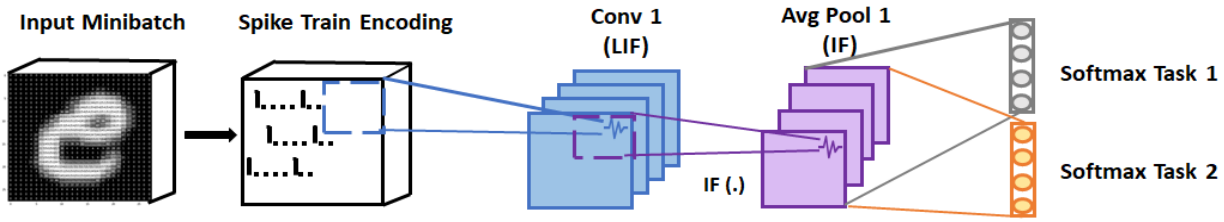


Fig. 2: The architecture of the Spiking CNN. Two softmax layers are used for transfer learning experiments which are replaced by one softmax for complete dataset experiments

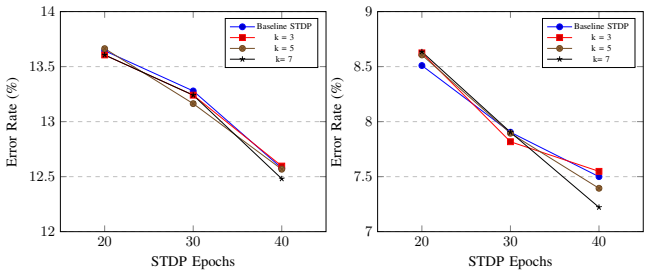


Fig. 3: Naive exponential decay performance a) test error rate for the classes (Task 1) which are used to train the convolutional layers using STDP and b) test error rate for the classes (Task 2) that are unseen to the convolutional layers

A. Naive weight decay

First, we detail an experiment with a naive exponential decay in the synapses, that is, a decay in weights proportional to their present value that occurs every k time steps during the training, where $k \in \{3, 5, 7\}$. We show this in the framework of transfer learning to set a baseline and motivate our further formulations. We denote this type of decay naive as it does not account for the temporal activations produced by the weights in a feature map but depends only on the values of weights across time. In addition to deciding on the interval, the decay rate that is a multiplicative factor determines the extent of the decay when the decay happens once in k time steps. These values of k and the decay rate were empirically determined so as to ensure that the decay does not overpower the weight updates by learning rule but complement it. The results are shown in Fig. 3. As observable, such a naive decay mechanism does not lead to improvement in the learning process.

B. Adaptive Threshold

A widely adopted homeostasis enabling method in fully connected SNNs is adaptive threshold, wherein the neuron's threshold is increased based on its activity. This enables other neurons learn a receptive field that is different. In spiking CNNs, the adaptive threshold is defined per activation map, with all neurons in an activation map bearing the same threshold [14]. As the threshold of the activation map is dictated by its activity, the threshold value at any time instant is indicative of its past activity. We perform an experiment to show that adaptive threshold in spiking CNNs applied in isolation does not lead to the desired homeostatic behavior.

Figure 4 shows Gaussian curves fitted to the distribution of thresholds of the activation maps after the completion of training as β_{thresh} in Equation 3 is varied. The increase in the mean of the distributions is a direct consequence but we are interested in the variance of the distribution which is observed to increase as well. Thus, adaptive threshold in isolation is likely not sufficient to encourage approximate equal activity of the activation maps over time.

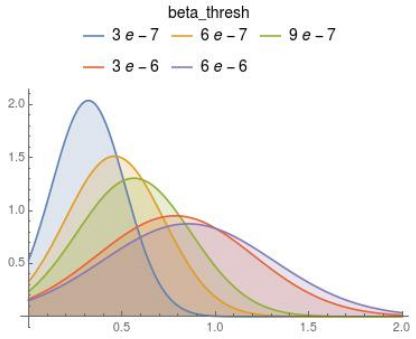


Fig. 4: Gaussian distribution fitted to the distribution of thresholds across activation maps on completion of training while varying β_{thresh}. Increase in β_{thresh} leads to an increase in both the mean and variance.

C. Identification of temporal parameters of interest

Motivated by the need to incorporate additional temporal parameters that dictate the decay mechanisms, we shed light on the temporal information captured by different parameters present in the network. First, we note that in fully connected spiking networks [12], weight decay was implemented such that it depends on the present activity of post trace and the cumulative past activity represented by the threshold of the neuron. Drawing inspiration, the threshold of the activation maps in spiking CNNs, which is indicative of the average post neuronal activity, was chosen to be the temporal metric for weight decay. In addition, it is important to note that in spiking CNNs, we are interested in the relative activity between the maps and not the absolute value of the threshold as the relative activity of the maps is representative of homeostasis. Thus, we use a metric henceforth referred to as normalized threshold $v_{th}^{normalized}$, wherein the normalization is done across the maps so as to capture the relative activity across maps.

Although a decay mechanism based on the $v_{th}^{normalized}$ may seem to be a reasonable metric that could provide homeostasis across activation maps, we observe empirically

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

that the $v_{th}^{normalized}$ varies marginally over time. This is due to the fact that the activity of the maps is reflective of the distribution of weights in the kernels to a good extent. This implies that a map that has a corresponding kernel with higher proportion of positive weights is likely to have a high threshold and dominate the normalized threshold. Any decay mechanism based on only the $v_{th}^{normalized}$ would only decay the weights of this dominant kernel with relatively no effect on the other maps. Also note that the dominant kernels cause a higher proportion of post neurons to spike, leading to higher frequency of weight updates. As the rate of decay is much smaller than the order of the initialized weights and the weight updates, there is effectively no regulation. This leads to the inference that $v_{th}^{normalized}$ does not provide the necessary temporal information of the post neuronal activity but only captures the information on the proportion of positive weights in the corresponding kernel.

v_{th}^{delta} is a measure of the activity of the activation map at the present time instant while $v_{th}^{normalized}$ is a measure of the proportion of the positive weights in the kernel over time. At any time instant, the product of v_{th}^{delta} and $v_{th}^{normalized}$ provides a metric that captures the temporal information of the post neuronal activity.

Bearing the above discussion in mind, we recall the STDP learning rule in Equation 2 in order to elaborate on the formulation of the decay mechanisms. We observe that if the exponent of the difference between t_{pre} and t_{post} is greater than the offset, the weight update would be in the positive direction. The latter weight dependent terms further modulate the weight update [13]. It is also important to note that the weight updates happen only when the post neuron fires. Cumulatively, we describe the scenarios under which we propose to decay offset and weights in Table I. Interpreting Equation 2 for a shared weight scenario, high $v_{th}^{normalized}$ and high v_{th}^{delta} in a map implies that the weights in the corresponding kernel are predominantly highly positive. This corresponds to the top left of the table wherein the weights are decayed. On the other hand, if their product is low, this would imply that the weights are highly negative, thus the offset is decayed to enable learning to take place. Additionally, if the v_{th}^{delta} is high but the $v_{th}^{normalized}$ is low, the learning rate is increased so as to enable the larger weight updates for less frequent weight updates. We use the expectation of $prod$, $\mathbb{E}[prod]$, as the metric to evaluate $high$ and low as we are interested in the relative activity. Building on the above insight on when the offset and weights have to be decayed, we now formulate the weight decay and offset decay.

4 cases of $prod$	$v_{th}^{normalized}$: High	$v_{th}^{normalized}$: Low
v_{th}^{delta}: High	Decay weights	Increase learning rate
v_{th}^{delta}: Low	Unlikely	Decay offset

TABLE I: Proposed decay mechanisms

We define $prod = v_{th}^{normalized} \times v_{th}^{delta}$.

At each time instant, $prod$ is evaluated for every map. Next, we identify the category under which each of the maps fall. All maps having the product higher than $\mathbb{E}[prod]$ fall in the upper left category in the table and undergo weight decay in their kernels while all maps lower than $\mathbb{E}[prod]$ fall in the lower right category of the table and undergo offset decay. We also define $diff_{map} = prod_{map} - \mathbb{E}[prod]$. Next, we use these temporal metrics to design weight and offset decay mechanisms.

A representative of the effect of the decay function is shown in Fig. 5, where the mean is set to 1. The blue and orange lines to depict the activity of maps at various time instants. For the given mean, both these maps are on the same side of the mean. From the formulation of the decay function, it is easy to observe that higher the activity is from the mean, higher the decay as seen in the Fig. 5. The blue map is initially higher in activity, analogous to the initialization of some weight kernels with higher proportion of positive weights. The decay of the blue map is much higher initially while at $\approx 100^{th}$ time step, the orange map's activity and decay is higher before they eventually reach similar activity which is representative of homeostasis.

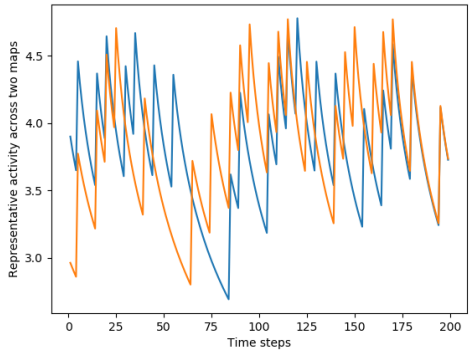


Fig. 5: A graphical representative of the effect of decay mechanisms on the activity of the maps

D. Relative Temporal Activation Driven Decay (R-Decay)

We model the weight and offset decay using the same type of function. We design the function to capture the essence of homeostasis such that higher the difference in activity on either side of the average of $prod$, higher is decay in weights or offset. Thus, a square term is used for the difference so as to capture either sides of the mean. The value of α, the decay constant, is decided empirically to complement the STDP weight updates and not overpower it. Thus, we have

$$\delta w = \alpha_{weights} * (exp(diff_{map}^2 - 1) * w$$
$$\delta offset = \alpha_{offset} * (exp(diff_{map}^2 - 1)) * offset$$

E. Range Adaptive Relative Temporal Activation Driven Decay (RAR-Decay)

On further observation from experiments, we notice that the range of $prod$ dynamically varies as the training progresses. The range of $prod$ for the dataset under consideration is significantly larger in its first few mini batches than its range later. In contrast, the decay does not modulate itself since the decay constant is fixed. Thus, the decay equation is modified

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

to adapt with the range of $prod$. The constant α in the equation is replaced by

$$\alpha_{weights} -> \beta_{weights}[diff_{max} - diff_{min}]^{-1}$$
$$\alpha_{offset} -> \beta_{offset}[diff_{max} - diff_{min}]^{-1}$$

where

$$diff_{max} = \max_{maps} diff_{map} \text{ and } diff_{min} = \min_{maps} diff_{map}$$

F. Relative Temporal Activation Driven Learning Rate (R-LR)

In addition to the above decay mechanisms, we propose a method to account for the case of low $v_{th}^{normalized}$ and high v_{th}^{delta} in the upper right scenario in Table I, which occurs when a kernel with a small proportion of weights is activated for a certain input. As this scenario implies that the frequency of weight updates to these kernels is low, we suggest an increase in learning rate. This is also motivated by the fact that although the offset decay helps learn better, it is important to note that learning happens only when the post neuron spikes. During initialization, if there are a large proportion of negative weights, the post neurons in corresponding maps spike less, due which there is a low frequency of updates to these negative weights. We propose to make the learning rate dependent on the inverse of the $v_{th}^{normalized}$. The STDP learning rate is changed to a variant of sigmoid function defined $beta$ and $gamma$ are determined to place the decay to modulate the learning rate in favorable regions. Thus, the learning rate adopted for each map η_{map}, with γ_1 and γ_2 being constants, is given by

$$\eta_{map} = \gamma_1 * (1 + \gamma_2 * \exp(\frac{-1}{v_{th}^{normalized}{}_{map}}))^{-1}$$

We finally depict our methodology in the form of a pseudo algorithm given by Algorithm 1.

Data: training and test examples
Result: Weights, Mean and std dev
$w_d \leftarrow$ [-1,1] according to Kaiming's initialization;
for x *in STDP training set* **do**
  for *t in time steps* **do**
    calculate the $v_{th}^{normalized}$ and v_{th}^{delta} for each map;
    STDP weight update with learning rate proportional to $(v_{th}^{normalized})^{-1}$;
    evaluate the $prod_{map}$, $diff_{map}$ for each map;
    for *map in maps* **do**
      if $diff_{map} > 0$ **then**
        $w- =$ $\beta_{weights}[diff_{max} - diff_{min}]^{-1} *$ $(exp(diff_{map}^2 - 1) * w$
      end
      else
        $offset- =$ $\beta_{offset}[diff_{max} - diff_{min}]^{-1} *$ $(exp(diff_{map}^2 - 1)) * offset$
      end
    end
  end
end
return $w_0, ..., w_{D+1}$

Algorithm 1: STDP coupled with decay mechanisms

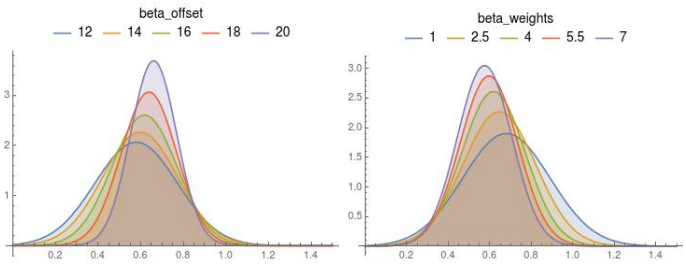
IV. Results

We perform experiments on the Extended MNIST dataset. The Extended MNIST dataset consists of 145,600 images from 26 balanced classes that represent the alphabets of the English language, consisting of both lower and upper case together within the same class. The dimension of the input space is 784 as a 2D square image. All images are used to train the softmax layer, in order to avoid overfitting, but a subset of the images are used to train the convolutional layers using STDP. This is due to a common observation that STDP learns its features from reasonably few examples and then saturates in its capacity, observed in both fully connected and convolutional architectures. The neuron time constant is set to $\tau = 0.9$ and the STDP learning rate used is 0.001. We perform the STDP training using minibatches of sizes of 200 and use 40 minibatches, constituting 8000 images, with the number of time steps being 25 and Poisson frequency of 200 Hz. The number of convolutional maps used are 8 unless mentioned otherwise. We make this choice due to the fact that an increase in number of maps also leads to an increase in the number of parameters mapping to the softmax layer trained using backpropagation. It is important to note that higher the number of parameters trained using backpropagation, more difficult it is to analyze the effect of variations in the training of convolutional layers using STDP and the decay mechanisms as the latter is overshadowed by the former. As we are keen on understanding the effect of decay mechanisms on the STDP training of the convolutional layers, we restrict the number of parameters trained using backpropagation to a minimum. This is also the reason for not adding any hidden layers trained using backpropagation. It is without doubt that the accuracy can be improved by adding a hidden layer or by increasing the number of maps, which we detail later. The softmax layer is trained with all the training examples using a batch size of 256 with Adam optimizer and cross entropy loss. For a given number of maps, the softmax training is done for a fixed number of epochs across the STDP variants used to maintain consistency in the contribution of backpropagation.

We use the standard method of initialization used for convolutional weight kernels in PyTorch, the Kaiming's initialization. These properties of random initialization tend to improve the baselines substantially for architectures such as the CNNs that perform localized projections [16]. We incorporate the above insight in this work, with Kaiming's initialization serving as the baseline for STDP while we show improvement with decay mechanisms over the baseline STDP.

First, we quantitatively show that the offset and weight decay mechanisms, coupled with adaptive threshold, lead to homeostatic behavior. In order to do so, we fit a Gaussian distribution to the thresholds across the maps. We observe that by

modulating the weights and offset decay constants, we are able to obtain a desired amount of homeostasis. Figure 6a shows the homeostatic behavior for a sweep across offset decay constant at a fixed weight decay constant ($\beta_{weights} = 4$) while Figure 6b shows the same across a sweep on the weight decay parameter with fixed offset decay constant ($\beta_{offset} = 16$). The steeper the distribution, the higher is the homeostatic behavior of the temporal activations. We observe that higher decay constants lead to higher homeostatic behavior and is highly controllable. However, akin to any regularization technique, there exists an optimal amount of regularization that leads to good generalization, which we measure using accuracy as a metric in the following experiments.



(a) (b)

Fig. 6: The distribution of thresholds across maps after STDP training modulated by a) offset and b) weights weights decay constant while the other is kept constant [(a) $\beta_{weights} = 4$ and b) $\beta_{offset} = 16$]. Increase in the decay constants change the variance substantially while marginally changing the mean

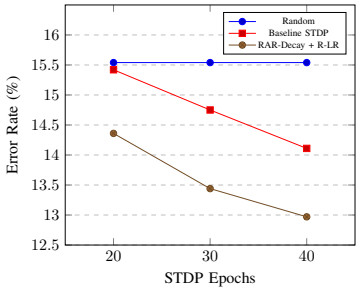
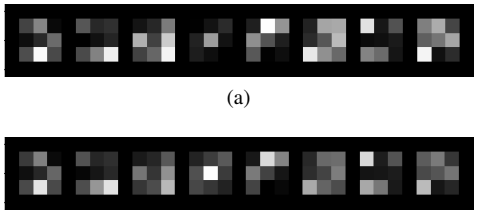


Fig. 7: Performance of RAR-Decay with R-LR as compared to STDP and random baselines over the complete dataset

For accuracy metrics, we first discuss our results on the full dataset across 26 classes. We consider a single softmax in this scenario with 26 output neurons. For the random baseline, we do not train the convolutional layer but only train the softmax using backpropagation. The results obtained with RAR-Decay with hyperparameters $\beta_{weights} = 4$, $\beta_{offset} = 16$, $\gamma_1 = 0.003$ and $\gamma_2 = 5$ are compared with baseline STDP and Random kernels in Fig. 7. We observe a that our decay mechanisms leads to improved learning with reduction in error rate up to 1.4% over the baseline STDP.

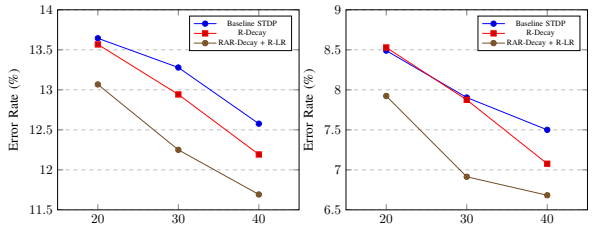
We show enhanced homeostatic behavior, obtained in addition to the improved accuracy, by plotting the weight kernels.



(a)

(b)

Fig. 8: Weight kernels when trained with a) baseline STDP; corresponding activation maps report Mean: 0.59, Std. Dev: 0.32 b) RAR-Decay + R-LR; corresponding activation maps report Mean: 0.62, Std. Dev: 0.15



(a) (b)

Fig. 9: Performance of our decay mechanisms a) test error rate for the classes (Task 1) which a used to train the convolutional layers using STDP and b) test error rate for the classes (Task 2) that are unseen to the convolutional layers

To quantify the same, we provide the mean and standard deviation of a Gaussian fitted to the thresholds of the maps. It is visibly noticeable in Fig. 8 that certain weight kernels have an improved contribution to the classification process. This is also corroborated by the reduced variance of the fitted Gaussian. This shows that the weight kernels produce activity that is relatively much closer to the mean among the maps, thus attaining the desired homeostatic behavior. This is complementary to the behavior observed on increasing β_{thresh} in isolation, where the variance increases.

Next, we conduct experiments to show that the decay mechanisms not only help learn the features for classes being trained better, these features are in fact transferable to classes that are not shown. We train the convolutional layers using STDP on only a subset of classes to demonstrate that the features learned are are useful for the data distribution in general. Thus, we keep a subset of data on which the convolutional layers are never trained on (referred to as Task 2) while they are trained on a different subset (referred to as Task 1). Fig. 9a) shows the performance on the subset that the convolutional layers are shown and for which the hyperparameters are adjusted while Fig. 9b) shows that the learned features serve the classes that are unseen as well. The hyperparameters used for R-Decay are $\alpha_{weights} = 750$ and $\alpha_{weights} = 1500$ while good performance in the RAR-Decay experiments are obtained with the same parameters used in the case of the full dataset. We note that the accuracy range difference between the two tasks is an artifact of the dataset.

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

Maps	Random		Baseline STDP		RAR-Decay + R-LR	
Error Rate (%)	T1 test	T2 Test	T1 Test	T2 Test	T1 Test	T2 Test
8	14.58	9.29	13.83	8.66	12.82	7.48
16	12.35	7.35	12.00	6.85	11.38	5.98
24	11.79	6.75	11.17	6.18	10.71	5.77

TABLE II: Error rate across sweep of number of maps

From Fig. 7 and Fig. 9, we observe the effect of the decay mechanisms on error rate. The temporal activation driven decays perform significantly better in comparison with baselines obtained with the naive weight decay in Fig. 3 as well as baseline STDP. As observed, the adaptation of the relative decay among maps to be in a certain range independent of the absolute value of the $prod$, as in the case of RAR-Decay as opposed to R-Decay, enables better feature learning. We observe that such a temporal decay mechanisms work better for both transfer learning and learning the task at hand.

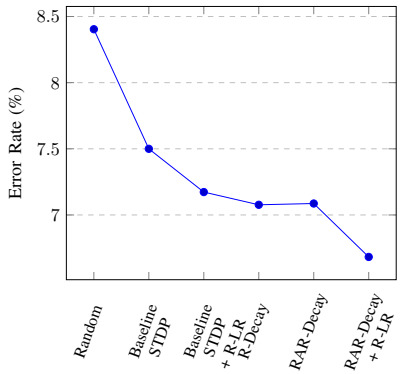


Fig. 10: Performance on the unseen classes (Task 2) with convolutional layer trained using STDP variants on a subset of classes (Task 1)

We also perform a sweep across the number of convolutional maps for completeness, keeping all other parameters as mentioned above and report our finding in Table II. As elucidated earlier, increasing maps leads to an increase in the number of parameters trained using backpropagation. In order to account for the same, and perform an iso-comparison across the effect of these backpropagation trained parameters, we compare across the number of epochs that achieve 90% of the saturation value. Finally, in order to quantitatively address the role of each variant of the decay mechanisms, we plot Task 2 performance when the convolutional features are trained on Task 1. We observe from Fig 10 that the learning rate being inversely proportional to $v_{th}^{normalized}$ together with RAR-Decay perform better together as compared to each individually.

V. Conclusion and Future Work

We show that spiking CNNs necessitate supporting mechanisms to adaptive threshold to ensure that all activation maps play a contributing role in the classification process. The decay mechanisms formulated in this work complement adaptive threshold, together achieving enhanced homeostatic behavior during training with unsupervised STDP. We show that the enhanced training translates to features that are transferable to the classes of the dataset that were not used to train the features. In order to scale to more complex datasets and deeper architectures, our future work entails the use of metrics such as Canonical Correlation Analysis (CCA) to better understand the effect of homeostasis on the feature representations [17]. Large number of parameters in the final layer trained using backpropagation obfuscate insights on the changes in the feature representation. CCA acts directly on the feature representations and proves useful in the above scenario.

Acknowledgement

This work was supported in part by the Center for Brain Inspired Computing (C-BRIC), one of the six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, by the Semiconductor Research Corporation, the National Science Foundation, Intel Corporation, the DoD Vannevar Bush Fellowship, and by the U.S. Army Research Laboratory and the U.K. Ministry of Defense under Agreement Number W911NF-16-3-0001.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1097–1105.

[2] e. a. Goodfellow, Ian, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 2672–2680.

[3] e. a. David Silver, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–503, 2016.

[4] R. Kempter, W. Gerstner, and J. L. van Hemmen, "Hebbian learning and spiking neurons," *Phys. Rev. E*, vol. 59, pp. 4498–4514, Apr 1999.

[5] W. Gerstner, A. K. Kreiter, H. Markram, and A. V. M. Herz, "Neural codes: Firing rates and beyond," *Proceedings of the National Academy of Sciences*, vol. 94, no. 24, pp. 12 740–12 741, 1997.

[6] P. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in Computational Neuroscience*, vol. 9, p. 99, 2015.

[7] P. Panda and K. Roy, "Unsupervised regenerative learning of hierarchical features in spiking deep networks for object recognition," in *2016 International Joint Conference on Neural Networks*, July, pp. 299–306.

[8] A. Tavanaei and A. S. Maida, "Multi-layer unsupervised learning in a spiking convolutional neural network," in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 2023–2030.

[9] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "Stdp-based spiking deep convolutional neural networks for object recognition," *Neural Networks*, vol. 99, pp. 56 – 67, 2018.

[10] J. C. Thiele, O. Bichler, and A. Dupret, "Event-based, timescale invariant unsupervised online deep learning with stdp," *Frontiers in Computational Neuroscience*, vol. 12, p. 46, 2018.

[11] P. Ferré, F. Mamalet, and S. J. Thorpe, "Unsupervised feature learning with winner-takes-all based stdp," *Frontiers in Computational Neuroscience*, vol. 12, p. 24, 2018.

[12] P. Panda, J. M. Allred, S. Ramanathan, and K. Roy, "Asp: Learning to forget with adaptive synaptic plasticity in spiking neural networks," *IEEE JETCAS*, vol. 8, no. 1, pp. 51–64, March 2018.

[13] C. Lee, G. Srinivasan, P. Panda, and K. Roy, "Deep spiking convolutional neural network trained with unsupervised spike-timing-dependent plasticity," *IEEE TCDS*, vol. 11, no. 3, pp. 384–394, Sep. 2019.

[14] G. Srinivasan and K. Roy, "Restocnet: Residual stochastic binary convolutional spiking neural network for memory-efficient neuromorphic computing," *Frontiers in Neuroscience*, vol. 13, p. 189, 2019.

[15] A. e. a. Paszke, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[16] B. Illing, W. Gerstner, and J. Brea, "Localized random projections challenge benchmarks for bio-plausible deep learning," 2019.

[17] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, 2019.