STDP-Based Pruning of Connections and Weight Quantization in Spiking Neural Networks for Energy-Efficient Recognition

Nitin Rathi, Priyadarshini Panda, *Student Member, IEEE*, and Kaushik Roy, *Fellow, IEEE*

***Abstract*—Spiking neural networks (SNNs) with a large number of weights and varied weight distribution can be difficult to implement in emerging in-memory computing hardware due to the limitations on crossbar size (implementing dot product), the constrained number of conductance states in non-CMOS devices and the power budget. We present a sparse SNN topology where noncritical connections are pruned to reduce the network size, and the remaining critical synapses are weight quantized to accommodate for limited conductance states. Pruning is based on the power law weight-dependent spike timing dependent plasticity model; synapses between pre- and post-neuron with high spike correlation are retained, whereas synapses with low correlation or uncorrelated spiking activity are pruned. The weights of the retained connections are quantized to the available number of conductance states. The process of pruning noncritical connections and quantizing the weights of critical synapses is performed at regular intervals during training. We evaluated our sparse and quantized network on MNIST dataset and on a subset of images from Caltech-101 dataset. The compressed topology achieved a classification accuracy of 90.1% (91.6%) on the MNIST (Caltech-101) dataset with 3.1X (2.2X) and 4X (2.6X) improvement in energy and area, respectively. The compressed topology is energy and area efficient while maintaining the same classification accuracy of a 2-layer fully connected SNN topology.**

***Index Terms*—Pruning, spike timing dependent plasticity (STDP), spiking neural network (SNN), unsupervised learning, weight quantization.**

I. Introduction

HUMAN brain consisting of 20 billion neurons and 200 trillion synapses is by far the most energy-efficient neuromorphic system with cognitive intelligence. The human brain consumes only ∼20W of power which is nine orders of magnitude lower compared to a computer simulating human brain activity in real time [1]. This had led to the inspiration and development of spiking neural networks (SNNs) which tries to mimic the behavior of human brain and process inputs in real time [2]. SNNs may provide an energy-efficient solution to perform neural computing. However, recent works have shown that to get reasonable accuracy compared to nonspiking artificial neural networks (nANNs), the complexity and size of SNNs is enormous. In [3] to improve the classification accuracy for MNIST dataset by 12%, the number of neurons in 2-layer SNN had to be increased by 64X. Lee *et al.* [4] achieved an average accuracy of 98.6% for MNIST dataset with two hidden layers consisting of 800 neurons in each layer. The quest of making SNNs larger and deeper for higher accuracy have compromised their energy efficiency and introduced challenges as mentioned below.

1) Large SNNs implemented on emerging crossbar structures [5], [24], [25] are limited by the crossbar size. Large crossbars suffer from supply voltage degradation, noise generated from process variations, and sneak paths [6], [7].
2) SNNs with numerous synapses involve higher number of computations making them slower and energy inefficient.

SNNs are driven by the synaptic events and the total computation, memory, communication, power, area, and speed scale with the number of synapses [2]. We propose a pruned and weight quantized SNN topology with self-taught spike timing dependent plasticity (STDP)-based learning. STDP, in turn, is also used to classify synapses as critical and noncritical. The noncritical synapses are pruned from the network, whereas the critical synapses are retained and weight quantized. Such pruning of connections and weight quantization can lead to their efficient implementations in emerging cross-bar arrays, such as resistive random access memories (R-RAMs) [8], [9], magnetic tunnel junctions [10], or domain-wall motion-based magnetic devices [11]. Such cross-bars, even though suitable for implementing efficient dot-products required for neural computing, are constrained in size, because of nonidealities, such as sneak paths, weight quantization, and parameter variations [6], [7]. The resulting sparse SNN can achieve $2-3X$ improvement in energy, $2-4X$ in area and $2-3X$ in testing speed.

Synaptic pruning is commonly observed during the development of human brain. The elimination of synapses begins at the age of two and continues till adulthood, when synaptic density stabilizes and is maintained until old age [12].

Manuscript received September 26, 2017; revised January 12, 2018; accepted March 6, 2018. Date of publication March 26, 2018; date of current version March 19, 2019. This work was supported in part by the National Science Foundation, in part by Intel Corporation, in part by the Vannevar Bush Faculty Fellowship, and in part by the C-BRIC, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) Program sponsored by DARPA. This paper was recommended by Associate Editor X. Li. *(Corresponding author: Nitin Rathi.)*

The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: rathi2@purdue.edu; pandap@purdue.edu; kaushik@purdue.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2018.2819366

From hardware implementation of neural networks, synapses are a costly resource and needs to be efficiently utilized for energy efficient learning. If synapses or connections are properly pruned, the performance decrease due to synaptic deletion is small compared to the energy savings [13]. This has motivated researchers to apply the technique of pruning [14] and weight quantization [15] to compress nANNs. Pruning and quantization performed on state-of-the-art network AlexNet trained for ImageNet dataset provided 7*X* benefit in energy efficiency along with 35*X* reduction in synaptic weight storage without any loss of accuracy [15]. Cun *et al.* [14] pruned the connections of an nANN trained using backpropagation based on the Hessian of the loss function. The number of parameters were reduced by a factor of two while maintaining the same test accuracy. The supervised learning algorithm in [16] pruned the hidden layer neurons with low dominance to reduce network size. The network achieved similar performance with 4*X* less parameters for Fisher Iris problem compared to other spiking networks. The idea of pruning is based on identifying parameters with small saliency, whose deletion will have minimal effect on the error. These networks were trained with supervised learning algorithms, but in SNNs with unsupervised training it is difficult to calculate such parameters since there is no such defined error function. In real world, obtaining unlabeled images for unsupervised learning is much easier than gathering labeled images for supervised learning. Iglesias *et al.* [17] designed an SNN with synapses characterized by activation levels. The activation level of a synapse changed according to the timing of the pre- and post-synaptic activity. The synapses with the lowest activation level after the network had stabilized were pruned to reduce network size. The process of pruning is applied only at the end and the remaining active synapses represent a small percentage of the overall connections. The novelty of our approach lies in self-taught STDP-based weight pruning where the connections to be pruned are decided based on their weights learned by the unsupervised STDP algorithm. Connections having STDP weights above a threshold are considered critical while others are temporarily pruned. The threshold is fixed before training and is referred as pruning threshold. The critical connections are weight quantized to further reduce network complexity. The resulting compressed topology is energy-efficient while maintaining accuracy and alleviates the issues that constrain the scalability of crossbar structure, leading to robust design of neuromorphic systems. The main contributions of this paper are mentioned below.

1) Online pruning based on the implicit correlation in neuronal activity resulting in a structured pruning methodology compared to simple thresholding.
2) Pruning the connections at regular intervals during training instead of just at the end to improve the training time.
3) The connections are pruned temporarily until the end of training to adapt to new training data. The low dominance connections are classified as non-critical and they become critical if new training data is introduced, therefore making the network scalable.

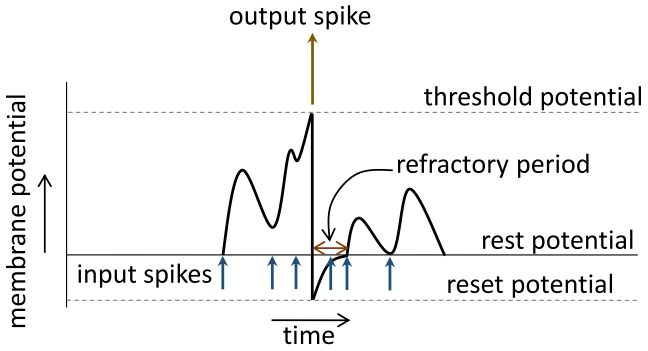

Fig. 1. LIF model of a single neuron's membrane potential dynamics in response to input spikes in SNN.

4) The quantization process is controlled by the underlying device technology implementing the synapse. The number of quantization levels depends on the available conductance states in the cross-bar arrays.

The rest of this paper is organized as follows. Section II provides background information on the neuron and the synapse models and the STDP learning algorithm employed in this paper. The network topology and the training and testing schemes are also briefly discussed. Section III presents the proposed compression techniques; STDP-based pruning and weight quantization and sharing. The experiments on the proposed topology are presented in Section IV. The results of the experiments are analyzed in Section V. The conclusions are drawn in Section VI. Section VI-A discusses the implication of pruning threshold on the tradeoff between accuracy and network size.

II. Background

A. Neuron & Synapse Model and STDP Learning

We employ the leaky-integrate-and-fire (LIF) model [3] to simulate the membrane potential dynamics of a neuron in our spiking network model. Fig. 1 shows the change in membrane potential of a single post-neuron in response to input spikes (blue arrows) from preneurons. The membrane potential increases at the onset of a spike and exponentially decays toward rest potential in the absence of spiking activity. The post-neuron fires or emits a spike when its potential crosses the threshold and immediately its potential is set to a reset value. After firing the post-neuron goes into a period of inactivity known as refractory period during which it is abstained from spiking, irrespective of input activity as shown in Fig. 1.

The connection between two neurons is termed a synapse and is modeled by the conductance change which is modulated by the synaptic weight (w). The synaptic weight between a pair of neurons increases (decreases) if the post-neuron fires after (before) the preneuron has fired. This phenomenon of synaptic plasticity, where the weight change is dependent on the inter spike timing of pre- and post-neuron is termed STDP. We adopt the power law weight-dependent STDP model, where the weight change is exponentially dependent on the spike timing difference of the pre- and post-neuron ($t_{\text{pre}} - t_{\text{post}}$) as well as the previous weight value [3].

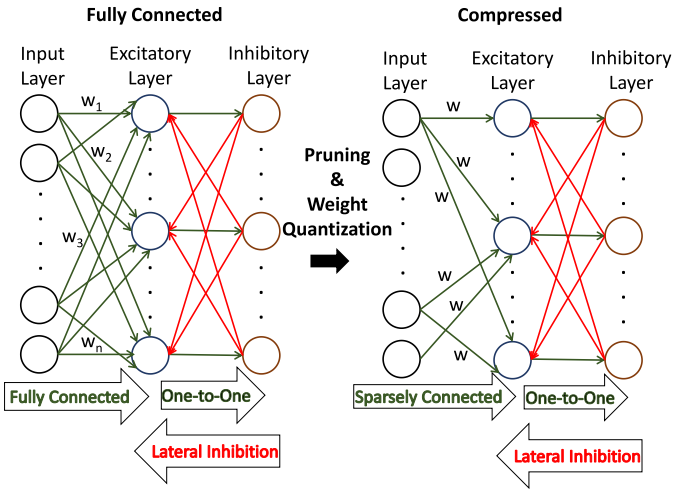

Fig. 2. SNN topology with lateral inhibition. Input to excitatory is fully connected which is later pruned. Excitatory to inhibitory is one-to-one connected, whereas inhibitory is backward connected to all the excitatory except the one it receives the connection from. Pruning is performed only on the input to excitatory connections.

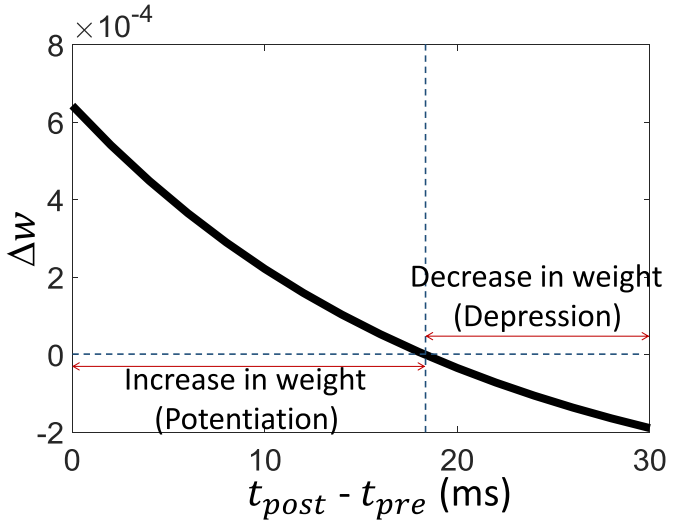

Fig. 3. Change in synaptic weight based on temporal correlation in pre- and post-synaptic spikes. ($\eta = 0.002$, $\tau = 20$ ms, offset $= 0.4$, $w_{\max} = 1$, $w = 0.5$, $\mu = 0.9$).

B. Network Topology

The SNN topology for this paper is shown in Fig. 2. It consists of input layer followed by excitatory and inhibitory layer. The input layer is fully connected to the excitatory layer, which in turn is one-to-one connected to the inhibitory layer. The number of neurons in the excitatory layer are varied to achieve better accuracy, whereas the number of neurons in the inhibitory layer is the same as the number in the excitatory layer. Each inhibitory neuron is backward connected to all the excitatory neurons except for the one from which it receives a connection from. Thus, the inhibitory layer provides lateral inhibition which discourages simultaneous firing of multiple excitatory neurons and promotes competition among them to learn different input features. The process of pruning and weight quantization is applied to the excitatory synapses to obtain the compressed topology as shown in Fig. 2. The fully connected topology serves as the baseline design and we compare the results of the compressed design with baseline.

To ensure similar firing rates for all neurons in the excitatory layer we employ an adaptive membrane threshold mechanism called homoeostasis [3]. The threshold potential is expressed as $V_{\text{thresh}} = V_t + \theta$, where V_t is a constant and θ is changed dynamically. θ increases every time a neuron fires and decays exponentially. If a neuron fires more often, then its threshold potential increases and it requires more inputs to fire again. This ensures that all neurons in the excitatory layer learn unique features and avoids few neurons from dominating the response pattern.

C. Training and Testing

The connections from input to excitatory layer are trained using the STDP weight update rule to classify an input pattern. The training is unsupervised as we do not use any labels to update the weights. The weight update is given by the formula described in Section III-A. The input image is converted into a Poisson spike train based on individual pixel intensities. The excitatory neurons are assigned a class/label based on their average spiking activity over all the images. During testing, the class prediction is inferred by averaging the response of all excitatory neurons per input. The class represented by the neurons with the highest spiking rate is predicted as the image label. The prediction is correct if the actual label matches the one predicted by the SNN. This is similar to the approach followed in [3].

III. Compression Techniques

In this section, we describe the two compression techniques (pruning and weight quantization) employed in this paper to convert the 2-layer fully connected SNN into a compact and sparse topology for digit and image recognition.

A. STDP-Based Pruning

STDP is widely used as an unsupervised Hebbian training algorithm for SNNs. STDP postulates that the strength of the synapse is dependent on the spike timing difference of the pre- and post-neuron. The power law weight update for an individual synapse is calculated as

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

where Δw is the change in weight, η is the learning rate, t_{pre} and t_{post} are the time instant of pre- and post-synaptic spikes, τ is the time constant, offset is a constant used for depression, $w_{\max}$ is the maximum constrained imposed on the synaptic weight, w is the previous weight value, μ is a constant which governs the exponential dependence on previous weight value. The weight update is positive (potentiation) if the post-neuron spikes immediately after the preneuron and negative (depression) if the spikes are far apart (Fig. 3). We employ STDP to train the excitatory synapses as well as to classify them as critical or noncritical. The synapses whose

weights do not increase for a set of inputs are likely to have not contributed toward learning and thus can be potential candidates for deletion. On the other hand, synapses with higher weights have most likely learned the input pattern and can be classified as critical (provided they were initialized with small weights). The characteristic features of the input is captured in connections with higher weights and are critical for correct classification. Thus, synapses with STDP trained weights ($w + \Delta w$) above pruning threshold are considered critical and all other synapses are marked as noncritical. The process of pruning and training is performed repeatedly by dividing the entire training set into multiple batches. After each batch the weights of all noncritical synapses are reduced to zero (they still remain in the network) and the network is trained with the next batch. The synapses with zero weight continue to participate in training, thus a noncritical synapse may become critical for different inputs. The process of reducing the weight to zero instead of eliminating the connection is essential for the network to learn the representation of inputs which appear in latter batches. The elimination of synapses will either make the network not learn the new representations or force the network to forget previous representations in order to learn new inputs. The process of retaining noncritical synapses with zero weight makes the network scalable. In the final training step when all the training images have been presented to the network any remaining noncritical connections are permanently removed from the network. The training starts with a fully connected network and the number of critical connections gradually decrease over time. At the end of training only the critical connections capturing the characteristic features of the inputs remain.

B. Weight Sharing and Quantization

The process of pruning reduces the overall connectivity, but as mentioned in Section I, SNNs with continuous weight values are difficult to implement in crossbar structures due to limitations on the number of available conductance states in devices implementing the synapse. Weight sharing and quantization discretizes the weights to the available number of conductance states. For example, network with 2-level weight quantization has only two values of weights: 0 (no connection) and w. All the synapses share the same weight (w) and the entire network can be represented as a sparse binary matrix. A 2-level weight quantized SNN can be implemented in crossbar architecture with a single fixed resistor [18], where w is the conductance of the resistor. The value of w is the average weight of all the critical connections trained using STDP. For example, we start with a network with n number of synapses and after training (with pruning) m critical synapses remain. The weights of the m critical synapses $(w_1, w_2, \ldots, w_m)$ are continuous and computed based on the STDP formula. The common weight value w is the average of w_1 to w_m. The average is calculated after each pruning step and all the critical connections share the same average weight (w_1 to w_m is replaced with w). Like pruning, the process of weight quantization and sharing is performed repeatedly after each training batch. The value of w changes at every quantization step and the final value is obtained after training the network for all the input batches. Similarly, the weights can be quantized to 3-levels: $0, w_1, w_2$, where $w_1(w_2)$ is the low (high) conductance value. The conductance values are computed by calculating the 50th percentile or the median weight of all the critical connections. The lower conductance value w_1 is the average of all weights between 0 and the median weight, w_2 is the average of rest of the weights. The critical synapses with weights between 0 and the median weight are assigned w_1. The critical synapses with weights between median weight and the maximum weight share the quantized value of w_2. The accuracy of the network is directly proportional to the number of quantization levels. The performance of the system improves with more number of conductance levels. In a quantized SNN most of the connections share the same weight which reduces the implementation complexity.

Fig. 4 summarizes the proposed algorithm for achieving a pruned and weight quantized SNN. The 2-layer untrained network is initialized with full connectivity from input to excitatory layer. The weights are randomly assigned from a uniform distribution. The training images are divided into N batches of equal number of images. The excitatory synapses are trained with STDP weight update rule for $M (M < N)$ training batches. The connections with current weights above the pruning threshold are classified as critical, rest of the connections are marked as noncritical. The noncritical connections are pruned by reducing their weights to zero. The weights of the critical synapses are quantized to the required number of conductance states. The pruned and quantized network is trained with STDP weight update rule for the next training batch. The process of pruning and quantization is performed at regular intervals for all the remaining training batches. The training ends when all the batches have been presented to the network. The first pruning and quantization step is delayed for M batches to ensure proper detection of critical connections and to mitigate the bias due to random initialization of weights. The randomly initialized synapses require more training images to capture the input characteristic features. Once the input features have been captured the pruning can be performed more often (after every batch). Once the critical connections are identified, they more or less remain the same during training. So, the first pruning step is very crucial and more than one training batch is needed to identify the critical synapses. The baseline design is trained in a similar fashion with no pruning and quantization. All the training images are presented in one batch and the weights are trained using STDP.

IV. Experimental Methodology

The proposed SNN topology is simulated in the open source spiking neuron simulator BRIAN implemented in Python [19]. BRIAN allows the modeling of biologically plausible neurons and synapses defined by differential equations. The parameters for the models are same as [3]. We tested our network for digit recognition on the MNIST dataset [20] and image recognition on a subset of images from the Caltech 101 dataset [21]. We propose two compression mechanisms: 1) pruning and 2) weight quantization. These mechanisms

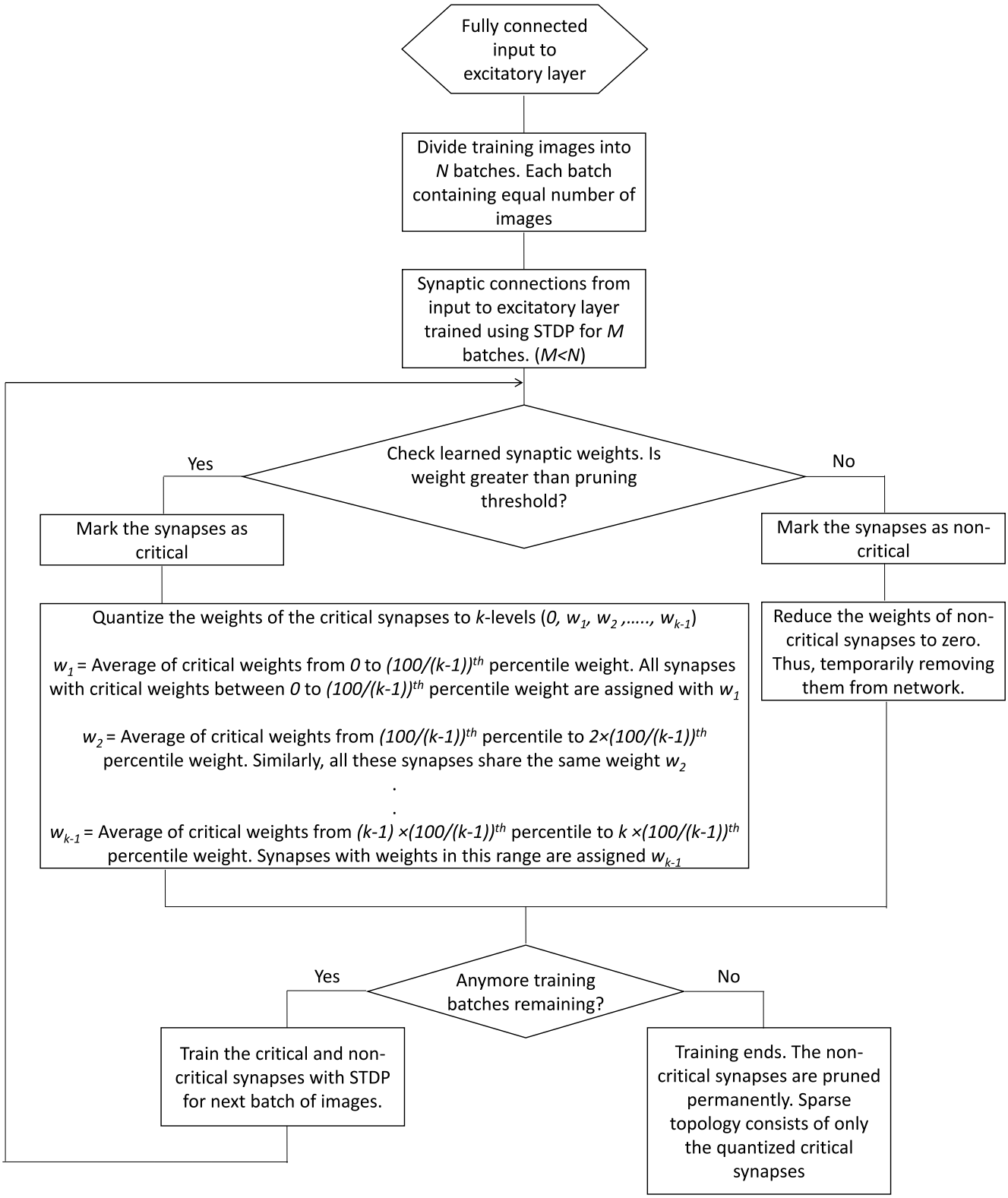

Fig. 4. Flowchart of the proposed algorithm for compressing SNN using pruning and weight quantization.

are applied on top of the baseline training algorithm. The baseline design is a fully connected network trained with STDP learning algorithm. We compare this design with a network trained in a similar fashion, but with compression techniques applied at appropriate intervals. Unlike nANNs which have pretrained models available like AlexNet, VGG Net, GoogLeNet, etc., SNNs do not have such standard pretrained models. Therefore, to compare our approach we train the baseline design in the absence of pruning and quantization.

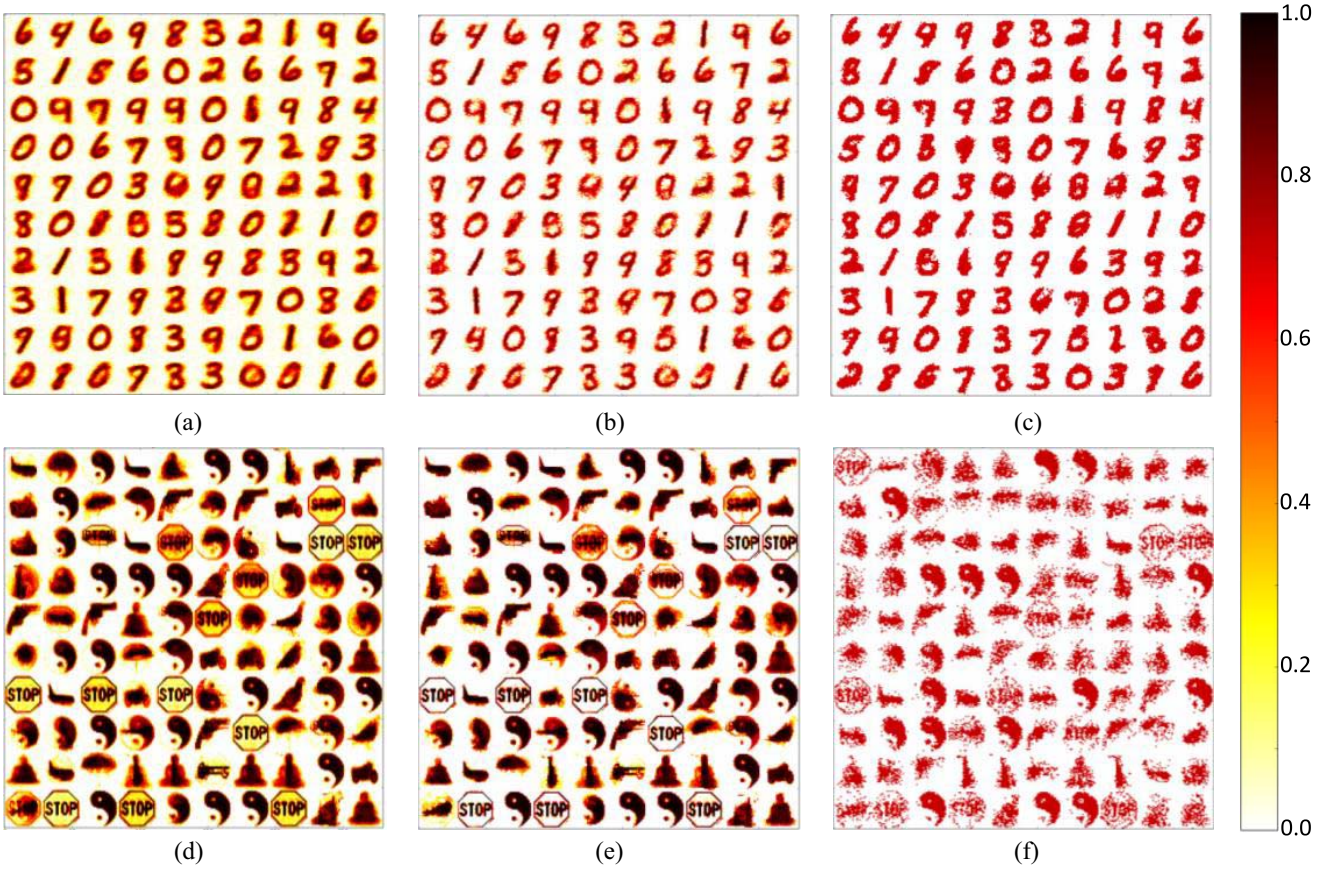

Fig. 5. Rearranged weights of the connections from input to excitatory for (a) MNIST baseline, (b) MNIST pruning, (c) MNIST pruning and quantization, (d) Caltech 101 baseline, (e) Caltech 101 pruning, and (f) Caltech 101 pruning and quantization.

A. MNIST Dataset

MNIST dataset contains $28 \times 28-$pixel sized grayscale images of digits 0-9. Thus, the input layer has 784 (28×28) neurons fully connected with 100 excitatory neurons. The dataset is divided into 60 000 training and 10 000 testing images. We further divide the 60 000 training images into batches of 5000 images ($N = 12$). The baseline design is trained with entire 60 000 images presented one after another. The compressed topology is initially trained for three training batches totaling 15 000 images ($M = 3$). STDP-based critical connections are weight quantized and the weights of the noncritical connections is reduced to zero. The pruned and quantized network is trained with the next training batch. The process of pruning and quantization is performed after every batch henceforth. The rearranged input to excitatory synaptic weights of the trained baseline topology with 100 excitatory neurons is shown in Fig. 5(a). Fig. 5(b) shows the rearranged synaptic weights of the same network compressed with a pruning threshold of 0.3 and having continuous weight values. The rearranged synaptic weights of the pruned and 2-level weight quantized network is shown in Fig. 5(c).

B. Caltech 101 Dataset

Caltech 101 dataset is a collection of images of objects belonging to 101 different categories. Each category consists of 40 to 800 of around $300 \times 200-$pixel sized RGB images. The dataset also provides annotations for the object in the image which we use to separate the object from the background. Unlike MNIST images, we preprocess the Caltech 101 images to obtain $28 \times 28-$pixel sized grayscale images. Maintaining the same image size across datasets ensures that we do not need to change the network parameters. Out of 101, we selected ten categories (yin yang, saxophone, stop sign, wrench, revolver, Buddha, airplanes, pigeon, motorbikes, umbrella) and randomly divided the total images in each category with 80% training and 20% testing images. Since each category has different number of images we create copies of images so that each category has similar number of training and testing images. This is necessary to avoid categories with more images to dominate the learning in the network. The preprocessing steps involved: converting the images to grayscale, averaging the pixels with Gaussian kernel of size 3×3 to suppress the noise and resizing the image to 28×28 pixels. All the preprocessing steps are performed using the OpenCV library [22] in python. The training set consists of 10 000 images with 1000 images per category. The 10 000 images are further divided into batches of 500 images ($N = 20$). The baseline fully connected design is trained with entire 10 000 images. The compressed topology is initially trained with ten training batches totaling 5000 images ($M = 10$). The critical connections are identified using STDP and weight quantized. The noncritical synapses are pruned. The pruned and quantized network is trained with all the remaining batches

with pruning and quantization performed after every training batch. Fig. 5(d)–(f) shows the rearranged synaptic weights for the baseline, pruned and weight quantized topologies, respectively. Compression is performed with pruning threshold of 0.2 and 2-level weight quantization.

V. Results and Analyses

In this section, we analyze the results and compare the performance of compressed topology with the baseline design. The results are evaluated based on different parameters like pruning threshold and number of excitatory neurons. The removal of connections during training is compared with training a sparse network, both having similar connectivity.

A. Comparison With Varying Pruning Threshold

The network connectivity is a strong function of the pruning threshold; higher the threshold, sparser is the network. The network connectivity is defined as the ratio of the actual number of connections to the total number of possible connections. The total number of possible connections with 100 excitatory neurons is 78 400 (784×100). The number of actual connections depend on the pruning steps. Fig. 6(a) and (b) shows the variation in final network connectivity with pruning threshold for MNIST and Caltech 101 datasets, respectively. The red dot in Fig. 6(a) and (b) with zero pruning threshold denotes the baseline design with no compression techniques applied. Ideally, the connectivity should be 1 since the connections are not pruned during training. The reduction in connectivity results from the inherent depression in the STDP learning rule. The further reduction in connectivity is achieved by increasing the pruning threshold. The compressed topologies are less sparse for low pruning threshold compared to baseline. This is due to weight quantization and sharing in early training stages. The shared weight is the average of all critical weights which is higher than almost half the critical weights. Thus, the average weight replaces half the STDP learned weights which were supposed to be much lower. This reduces the effect of inherent STDP depression on these synapses and reduces the probability of their removal. Fig. 6(c) and (d) shows the test accuracy for different network connectivity for MNIST and Caltech 101 datasets, respectively. The baseline topology has an accuracy of 81.6% (MNIST) and 84.2% (Caltech 101) which is consistent with the results shown in [3]. The highest classification accuracy achieved for the compressed topology is 79.5% (MNIST) and 82.8% (Caltech 101). The accuracy degrades slightly compared to baseline but at the same time there is immense drop in network connectivity. The compressed topology is 75% (36%) sparser than the baseline topology for MNIST (Caltech 101) dataset. The accuracy drops for high network connectivity for the weight quantized networks. In quantized networks most of the connections share the same weight values and as the connectivity increases the synapses become more alike. This introduces confusion in the network as the spiking activity for different classes become similar. This is not observed in networks with continuous weights because the individual weight values are different which makes the spiking activity of various classes differ from one another. Therefore, the quantized networks have to be sparse to achieve higher classification accuracy.

B. Comparison With Varying Number of Neurons

The change in classification accuracy with the number of excitatory neurons for MNIST and Caltech 101 datasets is shown in Fig. 6(e) and (f), respectively. The network is trained with a pruning threshold of 0.15(0.10) for MNIST (Caltech-101) dataset and the weights are quantized to 3-levels. These parameters correspond to the optimal tradeoff between accuracy and energy as discussed in Section VI-A. The baseline design with 6400 neurons achieved an accuracy of 93.2% for MNIST and 94.2% for Caltech 101 datasets. The pruned topology with similar number of neurons achieved an accuracy of 91.5% with 8% connectivity and 92.8% with 12% connectivity for MNIST and Caltech 101 datasets, respectively.

C. Pruning While Training

The objective of pruning is to increase the sparsity in the network. This can be achieved in multiple ways: removing connections during training of a fully connected network, training a sparse network or removing connections at the end of training. In first case the connections are removed systematically based on some parameters. The second approach is performed by randomly removing connections from a fully connected topology to produce a sparse network. The removal of connections at the end of training identifies low weight connections and prunes them from the network. This results in a network with similar sparseness but the network is no longer trained after the pruning step. Nevertheless, in all the cases the final trained network connectivity is same. Fig. 7(a) and (b) shows the classification accuracy with varying network connectivity for all three approaches for MNIST and Caltech 101 datasets, respectively. The networks with initial pruning and pruning at the end are trained similar to baseline with no compression techniques. The pruning while training is the approach followed in rest of this paper, where pruning is performed at regular intervals during training. The results for all the networks are shown for continuous weight distribution. Pruning the connections during training performs better since only the noncritical connections are removed. The network with initial sparsity is constructed by randomly removing connections. This shows that STDP successfully identifies the noncritical connections. Though pruning at the end of training removes some of the noncritical connections, the network's accuracy is lower compared to the proposed approach for highly sparse network. The absence of training after pruning and a single pruning step may be attributed for the reduction in classification accuracy.

D. Reduction in Spike Count or Energy

The decrease in connectivity due to pruning leads to reduced spiking activity in the excitatory layer. The active power of an SNN is proportional to the firing activity in the network [2]. Thus, the energy can be quantified as the reduction in spike count of excitatory neurons during testing. Fig. 8 shows the normalized reduction in spiking activity

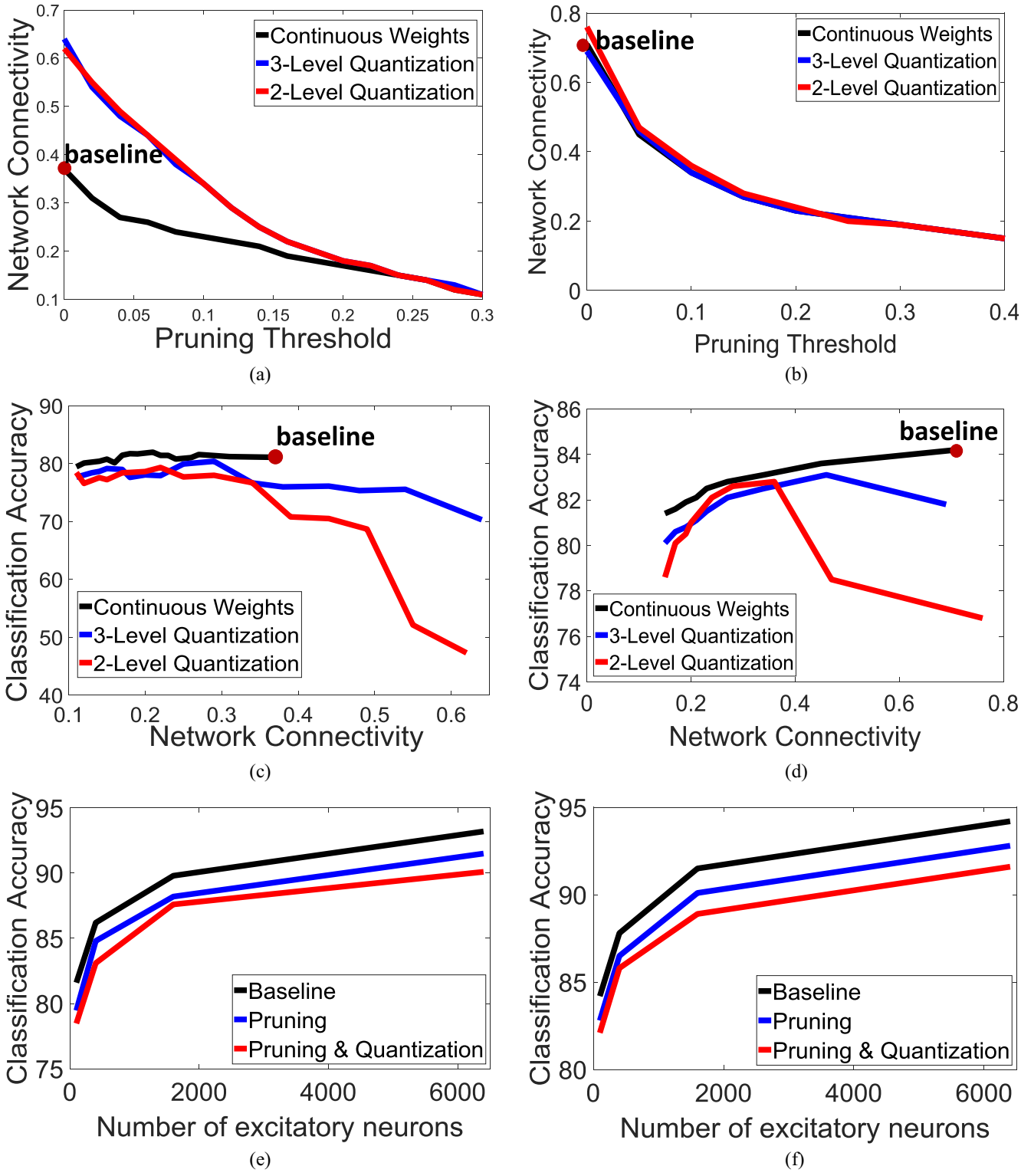

Fig. 6. Variation in network connectivity with pruning threshold for (a) MNIST and (b) Caltech 101. Classification accuracy for different network connectivity for (c) MNIST and (d) Caltech 101. Classification accuracy for different number of excitatory neurons for (e) MNIST and (f) Caltech 101.

or energy for compressed topology with respect to baseline. The pruned topology shows 3.1*X* and 2.2*X* improvement in energy, whereas the 2-level weight quantized network achieves 2.4*X* and 1.92*X* improvement for MNIST and Caltech101 datasets, respectively. The compressed topology may achieve additional energy benefits from implementation in crossbar structures with low power devices. The emerging post-CMOS devices like MTJ, R-RAM and domain wall motion-based devices consume very low power in idle state due to elimination of leakage. But these devices have limited number of programmable conductance states. The compressed topology quantized to the available number of conductance states can reap the energy benefits provided by these devices. The baseline design with continuous weight distribution is difficult to

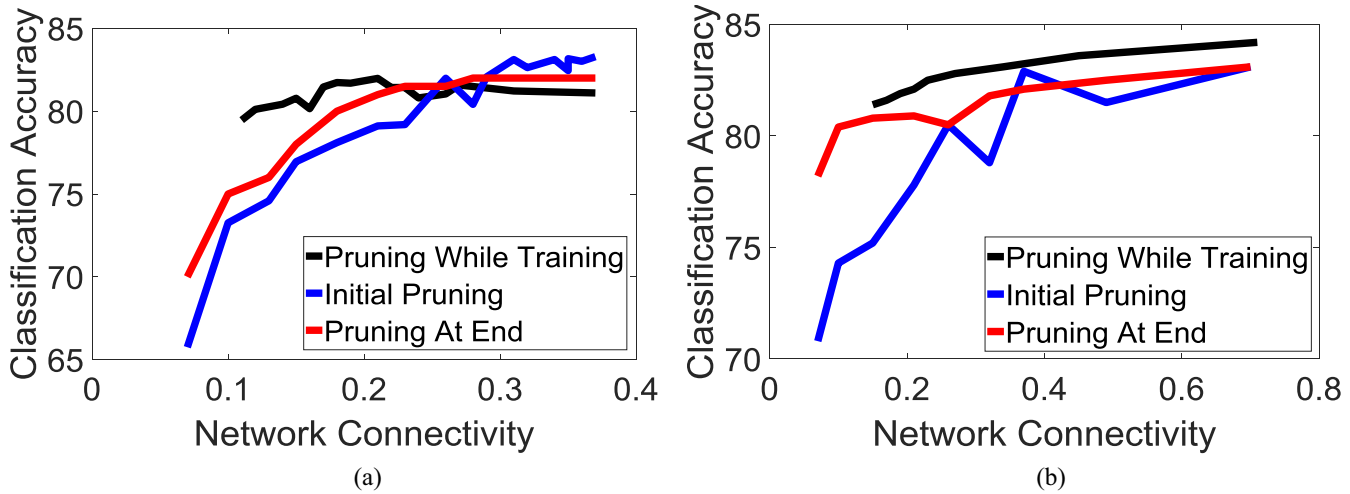

Fig. 7. Classification accuracy for different network sparsity achieved by pruning the connections during training, before training and after training for (a) MNIST and (b) Caltech 101.

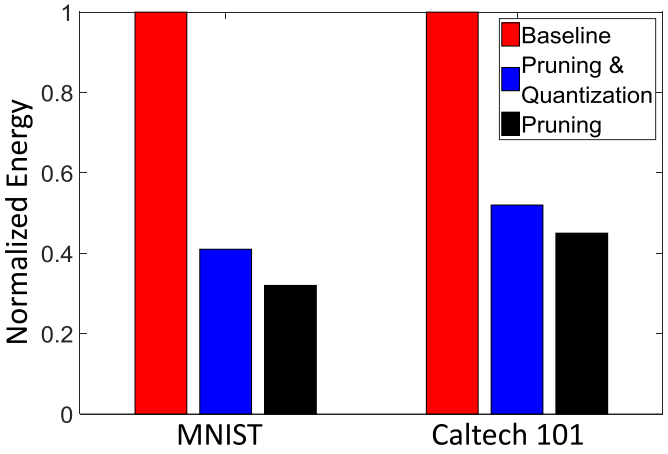

Fig. 8. Normalized improvement in energy with pruning and weight quantization compared to baseline topology.

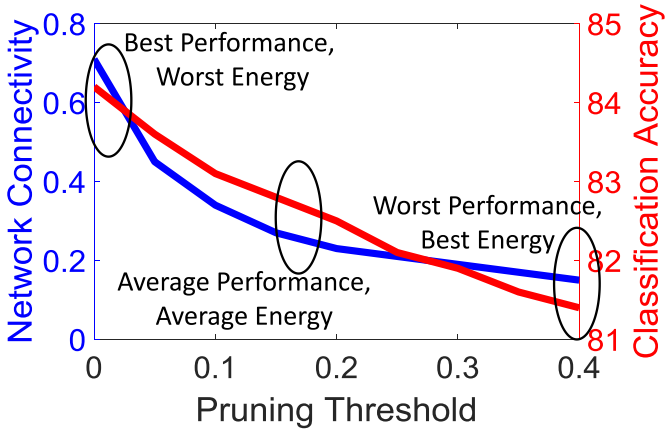

Fig. 9. Network connectivity and corresponding classification accuracy achieved with varying pruning threshold for a 100-neuron network with continuous weights and trained on a subset of categories from CALTECH-101 dataset.

implement with these devices. The introduction of sparseness also reduces the area of the cross-bar arrays. The number of devices in the cross-bar is proportional to the number of connections. Pruning threshold of 0.15(0.10) for MNIST (Caltech-101) dataset results in $4X(2.6X)$ reduction in number of connections with 0.6%(1.1%) drop in accuracy. The reduction in number of connections can be directly proportional to the area benefits if the cross-bar arrays are arranged efficiently. Therefore, our proposed approach results in $4X(2.6X)$ area reduction for MNIST (Caltech-101) dataset with minimal drop in accuracy.

VI. Conclusion

In this paper, we propose two compression techniques, pruning and weight quantization to compress SNNs. Compressed SNNs not only provide energy benefits but also mitigate the issue of limited programmable conductance states of post-CMOS devices for neuromorphic implementation. The novelty of our approach lies in fact that STDP learning rule is used to decide the network pruning and the weights of the critical connections are quantized to specific levels depending on device and technology requirements. The compressed topology is compared with the 2-layer fully connected topology for digit recognition with MNIST dataset and image recognition with Caltech 101 dataset. The proposed topology achieves $3.1X$ and $2.2X$ improvement in energy for MNIST and Caltech 101 datasets, respectively, compared to baseline fully connected SNN. The optimal compression parameters like pruning threshold and weight quantization levels are decided by performing multiple experiments with different images. Additionally, it is worth mentioning that the proposed topology reduced the training time by $3X$ and $2X$ for MNIST and Caltech 101 datasets, respectively, by achieving faster training convergence.

A. Discussion

Our results show that pruning and quantization can effectively reduce the number of connections during training in an SNN with minimal loss in accuracy. The process of pruning is controlled by the critical parameter "pruning threshold" and the weight quantization step requires to make a proper

judgement on the number of quantization levels. The number of quantization levels depend on the number of programmable conductance states available in the device technology implementing the synapse. Modern memristive cross-bars have shown 16 robust conductance states [23]. The accuracy of the network increases with more quantization levels and the best performance is achieved with continuous weights as shown in Fig. 6(a)–(f). To simplify our experiments, we considered only two and three level weight quantization along with continuous weights. The choice of pruning threshold is not as straightforward as it needs to consider the tradeoff between accuracy and reducing connections. Fig. 9 shows this tradeoff with varying pruning threshold. To the left, the pruning threshold is low resulting in dense network connectivity with high accuracy. To the right, the pruning threshold is high providing more area and energy benefits at the cost of accuracy degradation. Thus, the choice of pruning threshold depends on the application's tolerance on accuracy loss and energy budget. The bio-inspired STDP pruning mechanism allows to perform low power classification tasks but we still have a long way to go in order to match the accuracy and power efficiency of the human visual system. In the future we would like to include other mechanisms on top of the compression techniques to further improve accuracy and energy.

REFERENCES

[1] D. S. Modha. (2017). *Introducing a Brain-Inspired Computer*. [Online]. Available: http://www.research.ibm.com/articles/brain-chip.shtml

[2] P. A. Merolla *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[3] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Front. Comput. Neurosci.*, vol. 9, p. 99, Aug. 2015.

[4] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Front. Neurosci.*, vol. 10, p. 508, Nov. 2016.

[5] P. Merolla *et al.*, "A digital neurosynaptic core using embedded crossbar memory with 45pJ per spike in 45nm," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, San Jose, CA, USA, 2011, pp. 1–4.

[6] B. Liu *et al.*, "EDA challenges for memristor-crossbar based neuromorphic computing," in *Proc. ACM 25th Ed. Great Lakes Symp. VLSI*, Pittsburgh, PA, USA, 2015, pp. 185–188.

[7] E. Linn, R. Rosezin, C. Kügeler, and R. Waser, "Complementary resistive switches for passive nanocrossbar memories," *Nat. Mater.*, vol. 9, no. 5, pp. 403–406, 2010.

[8] M. Hu *et al.*, "Memristor crossbar-based neuromorphic computing system: A case study," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 10, pp. 1864–1878, Oct. 2014.

[9] S. Park *et al.*, "RRAM-based synapse for neuromorphic system with pattern recognition function," in *Proc. IEEE Int. Electron Devices Meeting (IEDM)*, San Francisco, CA, USA, 2012, pp. 10.2.1–10.2.4.

[10] P. Krzysteczko, J. Münchenberger, M. Schäfers, G. Reiss, and A. Thomas, "The memristive magnetic tunnel junction as a nanoscopic synapse-neuron system," *Adv. Mater.*, vol. 24, no. 6, pp. 762–766, 2012.

[11] M. Sharad, C. Augustine, G. Panagopoulos, and K. Roy, "Spin-based neuron model with domain-wall magnets as synapse," *IEEE Trans. Nanotechnol.*, vol. 11, no. 4, pp. 843–853, Jul. 2012.

[12] P. R. Huttenlocher, "Synaptic density in human frontal cortex—Developmental changes and effects of aging," *Brain Res.*, vol. 163, no. 2, pp. 195–205, 1979.

[13] G. Chechik, I. Meilijson, and E. Ruppin, "Synaptic pruning in development: A computational account," *Neural Comput.*, vol. 10, no. 7, pp. 1759–1777, 1998.

[14] Y. L. Cun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Advances in Neural Information Processing Systems 2*. San Mateo, CA, USA: Morgan Kaufmann, 1990, pp. 598–605. [Online]. Available: http://papers.nips.cc/paper/250-optimal-brain-damage.pdf

[15] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015.

[16] S. Dora, S. Sundaram, and N. Sundararajan, "A two stage learning algorithm for a growing-pruning spiking neural network for pattern classification problems," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Killarney, Ireland, 2015, pp. 1–7.

[17] J. Iglesias, J. Eriksson, F. Grize, M. Tomassini, and A. E. Villa, "Dynamics of pruning in simulated large-scale spiking neural networks," *Biosystems*, vol. 79, nos. 1–3, pp. 11–20, 2005.

[18] H. P. Graf *et al.*, "VLSI implementation of a neural network memory with several hundreds of neurons," *AIP Conf. Proc.*, vol. 151, no. 1, pp. 182–187, 1986.

[19] D. Goodman and R. Brette, "Brian: A simulator for spiking neural networks in Python," *Front. Neuroinformat.*, vol. 2, p. 5, Nov. 2008.

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[21] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006.

[22] G. Bradski, "The openCV library," *J. Softw. Tools Prof. Program.*, vol. 25, no. 11, pp. 120–123, 2000.

[23] E. J. Merced-Grafals, N. Dávila, N. Ge, R. S. Williams, and J. P. Strachan, "Repeatable, accurate, and high speed multi-level programming of memristor 1T1R arrays for power efficient analog computing applications," *Nanotechnology*, vol. 27, no. 36, 2016, Art. no. 365202.

[24] Y.-P. Lin, C. H. Bennett, T. Cabaret, D. Vodenicarevic, D. Chabi, D. Querlioz, B. Jousselme, V. Derycke, and J.-O. Klein, "Physical realization of a supervised learning system built with organic memristive synapses," *Scientific reports*, vol. 6, p. 31932, 2016.

[25] A. Ankit, A. Sengupta, P. Panda, and K. Roy, "Resparc: A reconfigurable and energy-efficient architecture with memristive crossbars for deep spiking neural networks," in *Proceedings of the 54th Annual Design Automation Conference 2017*. ACM, 2017, p. 27.

Nitin Rathi received the B.Tech. degree in electronics and communication engineering from the West Bengal University of Technology, Kolkata, India, in 2013. He is currently pursuing the Ph.D. degree in electrical and computer engineering with Purdue University, West Lafayette, IN, USA.

His current research interests include designing architecture and algorithms for low-power neuromorphic computing.

Priyadarshini Panda (S'16) received the B.E. degree in electrical and electronics engineering and the M.Sc. degree in physics from the Birla Institute of Technology and Science, Pilani, India, in 2013. She is currently pursuing the Ph.D. degree in electrical and computer engineering with Purdue University, West Lafayette, IN, USA.

Her current research interest includes low-power neuromorphic computing: energy-efficient realization of neural networks (spiking/nonspiking in deep learning context) using novel architectures and algorithms.

Kaushik Roy (F'01) received the B.Tech. degree in electronics and electrical communications engineering from IIT Kharagpur, Kharagpur, India and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Champaign, IL, USA, in 1990.

He was with the Semiconductor Process and Design Center, Texas Instruments Incorporated, Dallas, TX, USA, from 1990 to 1993, where he was involved in FPGA architecture development and low-power circuit design. He joined the Faculty of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, in 1993, where he is currently Edward G. Tiedemann Jr. Distinguished Professor. He has authored over 600 papers in refereed journals and conferences, holds 15 patents, graduated 75 Ph.D. students, and has co-authored two books on Low Power CMOS Very Large Scale Integration Design (Wiley and McGraw Hill). His current research interests include neuromorphic and cognitive computing, spintronics, device-circuit codesign for nanoscale silicon and nonsilicon technologies, low-power electronics for portable computing and wireless communications, and new computing models enabled by emerging technologies.