Deep Spiking Convolutional Neural Network Trained with Unsupervised Spike Timing Dependent Plasticity

Chankyu Lee, Gopalakrishnan Srinivasan, Student Member, IEEE, Priyadarshini Panda, Student Member, IEEE, and Kaushik Roy, Fellow, IEEE

Abstract—Spiking Neural Networks (SNNs) have emerged as a promising brain inspired neuromorphic-computing paradigm for cognitive system design due to their inherent event-driven processing capability. The fully-connected shallow SNNs typically used for pattern recognition require large number of trainable parameters to achieve competitive classification accuracy. In this work, we propose a deep Spiking Convolutional Neural Network (SpiCNN) composed of a hierarchy of stacked convolutional layers followed by a spatial-pooling layer and a final fully-connected layer. The network is populated with biologically plausible Leaky-Integrate-and-Fire (LIF) neurons interconnected by shared synaptic weight kernels. We train convolutional kernels layer-by-layer in an unsupervised manner using Spike Timing Dependent Plasticity (STDP) that enables them to self-learn characteristic features making up the input patterns. In order to further improve the feature learning efficiency, we propose using smaller $3 \times 3$ kernels trained using STDP-based synaptic weight updates performed over a mini-batch of input patterns. Our deep SpiCNN, consisting of two convolutional layers trained using the unsupervised convolutional STDP learning methodology, achieved classification accuracies of 91.1% and 97.6% respectively for inferring handwritten digits from the MNIST dataset and a subset of natural images from the Caltech dataset.

Index Terms—Spiking Neural Network, Convolutional Neural Network, Spike Timing Dependent Plasticity, Unsupervised Learning, Poisson-distributed Spike Encoding, Leaky-Integrate-and-Fire Neuron.

I. INTRODUCTION

BIOLOGICALLY inspired neuromorphic computing models are widely being explored in an effort to mimic the computational efficiency of the human brain in performing classification, recognition, and decision making among other tasks. Spiking Neural Networks (SNNs), popularly regarded as the third generation of neural networks, have garnered significant research interest because of their ability to closely emulate certain facets of computations performed by the human brain. The high bio-fidelity and intrinsic sparse event-driven processing capability render SNNs an ideal neuromorphic-computing paradigm for realizing energy-efficient hardware [1]–[3] with on-chip intelligence for classification and recognition applications. Spike Timing Dependent Plasticity (STDP), a prominent mechanism responsible for mammalian brain development [4], is typically used for the unsupervised training of SNNs.

Recently, SNNs widely explored for unsupervised pattern recognition consist of input neurons fully-connected by plastic synapses to a layer of excitatory (output) neurons [5]–[7]. However, the two-layered SNN topology necessitates significantly larger number of excitatory neurons to attain competitive classification accuracy. For instance, researchers in [5] showed that an SNN of 6400 excitatory neurons is needed to achieve a classification accuracy of 95% for MNIST digit recognition. This is because each excitatory neuron learns the complete structure of an image pattern. Consequently, more excitatory neurons are required to self-learn varying representations of different classes of input patterns. The ensuing rise in the number of synaptic weights increases the training complexity of the SNN. Furthermore, the two-layered SNN suffers from scalability issues with increasing number of output classes. In order to build an intelligent machine with improved scalability and reduced trainable parameters, it is imperative to devise a hierarchical SNN topology capable of extracting high-level features embedded in an image pattern and sharing learned features across different classes of patterns.

To this effect, we propose a deep Spiking Convolutional Neural Network (SpiCNN) composed of an input layer followed by a hierarchy of stacked convolutional layers for input feature extraction, a spatial-pooling layer for dimensionality reduction, and a fully-connected layer for final classification. Our proposal is inspired by the deep learning networks that exhibit state-of-the-art classification accuracies across a wide range of pattern recognition tasks while occasionally surpassing human performance [8], [9]. We train the convolutional weight kernels interconnecting consecutive layers in a sequential manner using STDP for self-learning distinctive features contained in the input patterns. The learned information is embedded in the average spiking rate of the neurons constituting the convolutional feature maps. We use the Poisson-distributed spike encoding mechanism for converting the average spiking rate of the neurons forming the input and convolutional feature maps to spike trains for training successive layers of SpiCNN. The chosen spike encoding scheme necessitates non-linear spiking neuronal model for competitive feature learning. Hence, we use the biologically plausible Leaky-Integrate-and-Fire (LIF) model [10] for the spiking neurons.
that offers rich non-linear behavior. However, recent efforts on training deep spiking networks using STDP explored temporal rank-order spike encoding scheme, where pixel intensity is represented by the relative order of incoming spikes in a network of integrate-and-fire neurons [11]–[14]. We note that Poisson-distributed spike encoding is more robust to intrinsic neuronal noise in comparison with the temporal rank-order scheme [15]. Moreover, rank-order coding incurs hardware overhead to modulate the weighted input spikes based on the respective order of firing. Poisson-distributed encoding, on the other hand, precludes this overhead since the individual spikes are statistically independent and information is encoded in the average neuronal firing rate. Further, the advances in nanotechnology have resulted in the emergence of neuro-mimetic devices capable of inherently mimicking dynamics of biological neurons and synapses [16]–[19]. We believe that the bio-inspired algorithms implemented using such emerging device technologies could pave the way for getting closer to the energy efficiency of the human brain. To the best of our knowledge, this work is the first demonstration of a multi-layered SNN using LIF spiking neurons and Poisson-distributed spike encoding scheme that is fully trained by spike-based STDP learning.

The application of STDP learning to a network of LIF neurons using Poisson-distributed spike encoding scheme has thus far been limited to shallow SNN topologies due to the challenge associated with propagating spikes across multiple levels of hierarchy that is critical for feature learning. In this work, we stack multiple convolutional layers and demonstrate effective feature learning by precisely modulating the network hyper-parameters including the LIF neuronal and STDP learning parameters. In an effort to minimize the sensitivity of SpiCNN to various hyper-parameters and improve the feature learning capability, we propose using smaller convolutional kernel sizes (for instance, 3×3 as opposed to the commonly used 5×5 or 7×7 kernel sizes) and training them using STDP-based mini-batch weight updates. The reduction in the number of trainable parameters offered by smaller kernel sizes together with STDP-based mini-batch weight updates enables the convolutional kernels to learn generalized features characterizing different input patterns. Our analysis indicates that smaller kernels learn prominent features and distributed internal representations across different layers, leading to improved classification accuracy. We comprehensively validate the efficacy of SpiCNN and the associated training methodology across different network topologies and kernel configurations using handwritten digits from the MNIST dataset [20] and natural images from the Caltech dataset [21].

Next, we provide the relevant background on SNNs and describe the proposed deep SpiCNN architecture and the STDP-based mini-batch unsupervised learning methodology. We subsequently outline the simulation framework and present experimental evidence illustrating the feature learning capability and robustness of SpiCNN. Finally, we highlight the benefits and trade-offs offered by our approach against recent works on deep SNNs.

II. SNN FUNDAMENTALS

A. Computational Model of Spiking Neuron and Synapse

The fundamental computing element of an SNNs is a spiking (post) neuron that is driven by a set of input (pre) neurons via plastic synapses. As formulated in (1), we use the Leaky-Integrate-and-Fire (LIF) model [10] to emulate the spiking neuronal dynamics.

\[
\tau_{\text{mem}} \frac{dV_{\text{mem}}}{dt} = -V_{\text{mem}} + W \cdot \delta(t - t_i)
\]

where \(V_{\text{mem}}\) is the neuronal membrane potential, \(W\) is the synaptic weight, \(t_i\) is the time instant at which an input pre-neuron spikes, and \(\tau_{\text{mem}}\) is the membrane time constant. The inputs are modeled as Dirac delta spike trains having unit magnitude at the time instant of a spike and zero elsewhere. The synaptic weights are constrained between \(-1\) and \(+1\). The input pre-neuronal spikes, denoted by \(\delta(t - t_i)\), are modulated by the interconnecting synaptic weights to produce a resultant current into the neuron. An LIF neuron integrates the input
current causing an increase in its membrane potential that
leaks exponentially once the input spikes are removed. It fires
an output spike if its potential exceeds an adaptive threshold.
The membrane potential is thereafter reset and the firing
threshold is raised. The various spiking activity regulatory
mechanisms including membrane potential decay and firing
threshold adaptation facilitate competitive feature learning.

B. Synaptic Plasticity

The strength of the synapses interconnecting a pair of pre-
and post-neurons is modulated using STDP, which postulates
that the synaptic strength (weight) varies exponentially with
the degree of timing correlation between the respective spike
patterns. In this work, we use the weight-dependent positive-
STDP rule that is illustrated in Fig. 1(a) and formulated below.

\[
\Delta W = \eta (e^{(t_{pre} - t_{post})/\tau} - \chi_{offset})(W_{\text{max}} - W)(W - W_{\text{min}})
\]

(2)

where \(\Delta W\) is the change in the synaptic weight, \(\eta\) is the
learning rate governing the amount of weight update, \(t_{pre}\)
and \(t_{post}\) are respectively the time instant of a pair of pre-
and post-neuronal spikes, \(\tau\) is the decay time constant, \(W\)
is the current synaptic weight, and \(W_{\text{max}}\) (\(W_{\text{min}}\)) is the
maximum (minimum) bound imposed on the synaptic weight.
As depicted in Fig. 1(a), the presented positive-STDP learning
rule uses only the positive timing window to measure the pre-
post spike timing difference. The synaptic weights are potenti-
dered/depressed by comparing the spike timing difference with
a threshold (i.e. \(\chi_{offset}\)). Synaptic potentiation is carried out
for strong temporal correlation between a pair of pre- and post-
spikes, i.e., if a pre-spike immediately causes the post-neuron
to fire as determined by \(\chi_{offset}\). On the contrary, synaptic
depression is carried out for larger spike-time differences.
The weight updates are applied only at the time instants of
post-synaptic spike and no weight change occurs at the time
instants of pre-synaptic spike. The change in synaptic weight
has a non-linear dependence on the current weight, constrained
between \(W_{\text{min}}\) of \(-1\) and \(W_{\text{max}}\) of \(+1\), to achieve a gradual
rise (decline) towards the maximum (minimum) bound. The
non-linear factor, specified by the product of \((W_{\text{max}} - W)\)
and \((W - W_{\text{min}})\) that simplifies to \((1 - W^2)\) and lies between
0 and 1, modulates the STDP-driven synaptic weight update
given by \((e^{(t_{pre} - t_{post})/\tau} - \chi_{offset})\). As illustrated in Fig. 1(b),
the non-linear factor ensures maximal STDP-driven update
if the current weight is closer to zero and minimal STDP-
driven update if the current weight is closer to the minimum
or maximum bound.

III. PROPOSED SNN ARCHITECTURE AND LEARNING
    METHODOLOGY

A. Deep Spiking Convolutional Neural Network (SpiCNN)

Our proposed SpiCNN (shown in Fig. 2) is composed of
a hierarchy of stacked convolutional (C) layers followed by a
spatial-pooling (S) layer and a final fully-connected (FC) layer.
The convolutional layers hierarchically extract characteristic
features from the complex input image patterns. For instance,
the first convolutional layer detects low-level features like
edges and corners while the successive layers extract high-
level features from the activation (feature) maps of the preceding
layer. This is accomplished by training the shared weight
kernels using the presented unsupervised convolutional STDP
learning methodology. The learned information is embedded
in the average spiking activity of the neurons forming the
convolutional feature maps.

Next, we have a spatial-pooling layer whose operation is
detailed using a fixed 2\times2 kernel with a stride of 2 pixels
at a time. Spatial-pooling operation reduces the dimension
of the convolutional feature maps while preserving the local
correlation between the constituent pixels. In this work, we
use average-pooling, which comprises each kernel weight of
1 and threshold of 0.25. During every stride of the kernel
over a convolutional feature map, an output spike is fired
by the corresponding neuron in the pooled feature map if
any of the 4 input pixels spikes. This, in effect, reduces the
dimension of the convolutional feature map by a factor of
two with minimal loss of spike information. Note that the
weight kernel used for spatial-pooling is not trainable and
is fixed a priori. Spatial-pooling has the following two-fold
advantages. First, it enhances the computational efficiency by
reducing the dimension of the convolutional feature maps.
Second, it renders the network invariant to slight distortions
and translation in the input patterns.

Finally, we have a fully-connected (output) layer containing
as many neurons as the number of classes for a given recogni-
from other input patterns. In SpiCNN, we allow the kernels in the remaining feature maps to learn different features dominating learning and facilitates the weight kernels housed old adaptation scheme effectively prevents few kernels from them exponentially over time. The proposed uniform thresh-

while decaying \( \varepsilon \) map fire, we increase the firing thresholds of all the neurons explained below. Whenever few neurons in a particular feature map fire, we increase the firing thresholds of all the neurons in the feature map by a constant value (\( \varepsilon \)) while decaying them exponentially over time. The proposed uniform threshold adaptation scheme effectively prevents few kernels from dominating learning and facilitates the weight kernels housed in the remaining feature maps to learn different features from other input patterns. In SpiCNN, we allow the kernels to have both positive and negative weights as illustrated in Fig. 3 for regulating the spiking activity of the neurons within a feature map. This precludes the need for explicit lateral inhibitory synaptic connections within a feature map that are typical in SNNs using only positive weights as demonstrated in [5], [14]. Even though negative weights are not biologically plausible, we incorporate them to achieve a reduction in the network complexity. These mechanisms collectively enable the weight kernel to self-learn unique input features. After one layer is trained, adjusted convolutional weights are frozen and firing thresholds of feature map are scaled by a constant factor (\( \beta \)) in order to increase spiking activities in the output feature map. We estimate the nonlinear transformation of input by feeding spikes from input through trained layers. Accordingly, following layer is trained by passing through regenerated spike events of nonlinear transformation of input pattern. This process is repeated until all the layers are trained.

**B. Unsupervised Convolutional STDP Learning Methodology**

We train the convolutional weights interconnecting successive pairs of layers of SpiCNN in a greedy layer-wise and unsupervised manner using STDP learning. At every time-step, the pre-neuronal spikes from the input feature maps are convolved with the respective weight kernels to generate a resultant current into the neurons constituting the output feature maps as depicted in Fig. 3. In the event of a post-neuronal spike, STDP-based updates are applied to the corresponding kernel weights. If several neurons making up an output feature map spike, as is typically observed, an average update is carried out on the shared kernel weights based on the respective pre- and post-neuronal spike times. In order to achieve efficient unsupervised learning, we propose uniform threshold adaptation across all neurons in a feature map as explained below. Whenever few neurons in a particular feature map fire, we increase the firing thresholds of all the neurons in the feature map by a constant value (\( \varepsilon \)) while decaying them exponentially over time. The proposed uniform threshold adaptation scheme effectively prevents few kernels from dominating learning and facilitates the weight kernels housed in the remaining feature maps to learn different features from other input patterns. In SpiCNN, we allow the kernels to have both positive and negative weights as illustrated in Fig. 3 for regulating the spiking activity of the neurons within a feature map. This precludes the need for explicit lateral inhibitory synaptic connections within a feature map that are typical in SNNs using only positive weights as demonstrated in [5], [14]. Even though negative weights are not biologically plausible, we incorporate them to achieve a reduction in the network complexity. These mechanisms collectively enable the weight kernel to self-learn unique input features. After one layer is trained, adjusted convolutional weights are frozen and firing thresholds of feature map are scaled by a constant factor (\( \beta \)) in order to increase spiking activities in the output feature map. We estimate the nonlinear transformation of input by feeding spikes from input through trained layers. Accordingly, following layer is trained by passing through regenerated spike events of nonlinear transformation of input pattern. This process is repeated until all the layers are trained.

The proposed greedy layer-wise unsupervised training methodology ensures that each layer receives sufficient input spikes to achieve efficient learning, thereby mitigating the issue of gradual decrease in spiking activity across layers that is inherent in deep SNNs [22]. However, extracting general characteristics of input patterns and sensitivity to hyper-parameters are a couple of key challenges that need to be addressed to achieve efficient learning. To this effect, we propose employing smaller \( 3 \times 3 \) weight kernels instead of commonly used \( 5 \times 5 \) or \( 7 \times 7 \) kernels for deep SNNs. Our experimental analysis shows that smaller kernels trained using STDP-based mini-batch weight updates self-learn general input representations, which causes them to extract prominent characteristic features across successive convolutional layers. This leads to learn distributed internal representations compared to larger kernels across hidden convolutional layers of SpiCNN, which enhances the recognition capability of the final fully-connected layer. Furthermore, the decrease in the number of trainable parameters as a result of using smaller kernels together with performing mini-batch weight updates renders the feature learning efficiency less sensitive to the various network hyper-parameters. In mini-batch training, we compute
the STDP-based weight updates individually for each input pattern in a randomly selected mini-batch of input patterns. We subsequently modify the kernels using weight updates averaged over the chosen mini-batch. Performing an average weight update (over a mini-batch) enables each kernel to extract features common to different classes of input patterns. Mini-batch training, in essence, causes general feature learning using fewer weight updates.

C. Supervised STDP Learning Methodology for final layer

Finally, we train the fully-connected layer in a supervised manner using the positive-STDP rule formulated in (2). For a given training pattern, the STDP-based weight updates are carried out on the output neuron that is pre-assigned to learn the particular input class. The firing thresholds of the remaining output neurons are momentarily raised to a high enough value that effectively prevents them from firing. The neuron that is supposed to learn the presented input pattern is guaranteed to spike at the beginning of the training period since it is initialized to a threshold of zero. Every time the neurons spike, we increase their threshold by a small amount while exponentially decaying over time in order to maintain spiking activities during a course of the training period. Threshold adaptation ensures that the frequency of synaptic weight updates is high at the beginning of the training period and is gradually lowered, leading to efficient learning. When more than one output neuron fire in the final layer, the gradual threshold adaptations and competitive inhibitions take place together to regulate spiking activities and accentuate them between the output neurons.

IV. SIMULATION FRAMEWORK

We evaluated the proposed SpiCNN using a MATLAB-based custom simulation framework. We pre-processed the original input image using the Laplacian of Gaussian (LoG) filter [23] to extract the contrasts among the image pixels and accentuate the edges. We truncated the LoG-filtered image pixels between 0 and 1, and subsequently converted them to Poisson-distributed spike trains whose firing rates are constrained between 0 and 400 Hz for training the convolutional layers. We used a reduced maximum Poisson firing rate of 200 Hz for training the final fully-connected layer. These input spike trains are kept active for a time period of 25 msec during training and 300 msec during inference assuming a simulation time-step of 1 msec. The synaptic weights are initialized randomly following a uniform distribution as formulated below.

\[
W_{j,j+1} \in U[-\sqrt{\frac{6}{n_j + n_{j+1}}}, \sqrt{\frac{6}{n_j + n_{j+1}}}] 
\]  

(3)

where \(W_{j,j+1}\) is the synaptic weight matrix interconnecting layers \(j\) and \(j + 1\), \(U[-k,k]\) denotes a normal distribution in the interval between \(-k\) and \(k\), and \(n_j\) and \(n_{j+1}\) are the number of neurons in layers \(j\) and \(j + 1\) respectively. Note that this weight initialization scheme is typically used in deep networks for breaking the symmetry between the units in each layer [24]. The neurons in the convolutional and fully-connected layers are modeled using Leaky-Integrate-and-Fire dynamics [10]. The STDP-based update on the synapse interconnecting a pair of pre- and post-neurons is implemented by generating an exponentially decaying trace integrating the pre-spike, and sampling it at the instant of a post-spike to update the synaptic weight. We trained the shared weight kernels using the presented unsupervised convolutional STDP learning methodology and validated its efficacy by evaluating the classification accuracy on an independent testing dataset. For a given test pattern, we accumulate the spike-count of the output neurons over a period of 300 msec and predict the test pattern to belong to the class represented by the output neuron with the highest spike count.

V. RESULTS

A. MNIST Digit Recognition

We demonstrate the unsupervised feature extraction capability of deep SpiCNN by training it to infer handwritten MNIST [20] digits. The MNIST dataset consists of 60K training images and 10K testing images, each 28x28 in dimension and encoded in grayscale format. We LoG-filtered the original MNIST image and subsequently converted it to a binary image using a pre-defined pixel threshold. The LoG-filtered pixel intensities lower than the threshold including the negative values are truncated to 0 while those above the threshold are clipped to 1. Fig. 4 shows few samples from the input datasets and the respective LoG-filtered binary and grayscale images.
TABLE I
SPICNN SIMULATION PARAMETERS.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>STDP Type</td>
<td>Nearest STDP</td>
</tr>
<tr>
<td>Synaptic Weight Range</td>
<td>[-1, 1]</td>
</tr>
<tr>
<td>Minimum Simulation Time-Step</td>
<td>1 msec</td>
</tr>
<tr>
<td>Decay Time Constant of Membrane Potential</td>
<td>100 msec</td>
</tr>
<tr>
<td>Decay Time Constant for Positive-STDP</td>
<td>2 msec</td>
</tr>
<tr>
<td>Decay Time Constant for Threshold</td>
<td>1000 msec</td>
</tr>
<tr>
<td>Training Duration</td>
<td>25 msec</td>
</tr>
<tr>
<td>Inference Duration</td>
<td>300 msec</td>
</tr>
<tr>
<td>Maximum Input Spiking Rate for Training</td>
<td>200 Hz – 400 Hz</td>
</tr>
<tr>
<td>Maximum Input Spiking Rate for Inference</td>
<td>200 Hz</td>
</tr>
<tr>
<td>Mini-batch Size</td>
<td>200 images</td>
</tr>
<tr>
<td>Convolutional Kernel Stride</td>
<td>1</td>
</tr>
<tr>
<td>Pooling Layer Stride</td>
<td>2</td>
</tr>
<tr>
<td>Neuronal Threshold Increase when Firing</td>
<td>1.5</td>
</tr>
<tr>
<td>Threshold Scaling Factor (β)</td>
<td>1.5</td>
</tr>
<tr>
<td>Lateral Inhibition Factor at Final Layer</td>
<td>0 – 5</td>
</tr>
</tbody>
</table>

It is important to note that the presented convolutional STDP learning methodology is capable of self-learning features both from the grayscale and binary images. Nevertheless, we begin our analysis by using binary images to determine the optimal network topology including the kernel size, number of kernels within a layer, and number of layers. We then train the optimal SpiCNN topology with both the grayscale and binary images, and present insights on how the choice of input encoding and spike encoding schemes impact the feature learning efficiency.

In our first experiment, we trained a shallow 1C-1S-FC SpiCNN, which is composed of a single convolutional layer (C) followed by a spatial-pooling (S) layer and a fully-connected (FC) layer, across a range of kernel sizes and number of kernels making up the convolutional layer using the parameters listed in Table I. Fig. 5(a) illustrates the general features including the horizontal, vertical, and diagonal edges acquired by the shared kernels, which is a testament to the efficacy of the presented convolutional learning methodology performing STDP-based mini-batch synaptic weight updates. We used a mini-batch size of 200 training images for enabling the convolutional kernels to self-learn shared features across different classes of input patterns. Our results (shown in Fig. 5(b)) indicate that the classification accuracy of the 1C-1S-FC SpiCNN, in general, increases with the number of kernels. Furthermore, we found that the larger 5 × 5 or 7 × 7 kernels outperform the 3 × 3 kernels by up to 2.0% in classification accuracy. This can be attributed to the ability of larger kernels to encode more features in such shallow topologies. The 1C-1S-FC SpiCNN with 36 5 × 5 kernels achieved a maximum classification accuracy of 89.5% on the MNIST testing dataset.

In an attempt to further enhance classification accuracy, we stacked multiple convolutional layers to form a deep SpiCNN. We explored a couple of deep SpiCNN configurations, namely, 2C-1S-FC and 3C-1S-FC SpiCNN, consisting of two and three stacked convolutional layers respectively with 16 feature maps per layer. Our experimental analysis showed that deep SpiCNN with 3 × 3 kernels yielded improved classification accuracy over the one with 5 × 5 kernels while also performing significantly better than that with the larger 7 × 7 kernels as illustrated in Fig. 5(c). This is in stark contrast to the trend observed for shallow SpiCNNs (refer to Fig. 5(b)). The improved recognition capability of deep SpiCNN using 3 × 3 kernels stems from their ability to learn prominent features and distributed internal representations across successive convolutional layers and the ensuing increase in hidden spiking activity, as illustrated in Fig. 5(b) and Fig. 5(c). The smaller 3x3 kernels can learn generalized representations of input data as a result of having fewer trainable parameters, while larger kernels with a greater number of trainable parameters learn specific features that are less common to overall classes of input patterns.

Given $\chi_{offset}$ is a threshold that is compared with the spike timing difference at the time instant of post-synaptic spikes, there is an optimal $\chi_{offset}$ value depending on kernel size to determine the sharpness of kernel shape and retain a certain amount of spiking activity. If $\chi_{offset}$ is set high, the convolutional kernel learns a sharper shape to incur a drastic decrease in spiking activity at the output feature map since they only detect a particular pattern. The inherent characteristic of the smaller kernel to learn generalized representations of input patterns allows the increase $\chi_{offset}$ to extract prominent features prevalent among different classes of input data. Conversely, a larger kernel should have lower $\chi_{offset}$ since it learns the specific features that produce a less
TABLE II
THE LEARNING PARAMETER $\chi_{offset}$.

<table>
<thead>
<tr>
<th>Network Topology</th>
<th>1C-1S-FC</th>
<th>2C-1S-FC</th>
<th>3C-1S-FC</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST Kernel Size</td>
<td>3x3</td>
<td>5x5</td>
<td>7x7</td>
</tr>
<tr>
<td></td>
<td>0.27</td>
<td>0.26</td>
<td>0.25</td>
</tr>
<tr>
<td>Caltch (Face/Motorbike) Kernel Size</td>
<td>3x3</td>
<td>5x5</td>
<td>7x7</td>
</tr>
<tr>
<td></td>
<td>0.32</td>
<td>0.27</td>
<td>0.23</td>
</tr>
</tbody>
</table>

Fig. 6. Feature maps at the output of every layer of a 2C-1S-FC SpiCNN for different kernel sizes.

Fig. 7. Normalized Spiking activity of feature maps at the output of every layer of a 2C-1S-FC SpiCNN for different kernel sizes for (a) MNIST handwritten digit and (b) Caltch (Face/Motorbike) datasets.

discernable output feature maps which drastically decreases the spiking activity across successive layers. Given that a certain amount of spiking activity at the input feature maps of fully-connected layer is needed to reasonably infer the class of the input patterns, we adjusted $\chi_{offset}$ as illustrated in Table II in order to match the output spiking activity of successive convolutional layers by considering the kernel sizes and depth of network configurations. As depicted in Fig. 7, the output spiking activities after the successive convolutional layers are similar for all 2C-1S-FC SpiCNN configurations, but the spiking activities at the output of the first convolutional layer depends on the convolutional kernel sizes. The results show that 3x3 and 5x5 kernels produces higher spiking activities than the 7x7 kernels at the output of the first convolutional layer, which indicates the degree of resultant features captured from the input patterns. In other words, the smaller kernel learns prominent features that are common
to overall output classes and constructs distributed internal representations across successive layers as illustrated in Fig. 6. Note that researchers in [25] have validated the efficacy of smaller 3x3 kernels in the traditional deep learning networks. Hence, the 2C-1S-FC SpiCNN with 3x3 kernels across both the convolutional layers containing the same number of feature maps offered the highest accuracy of 91.1%. Last, we note that the classification accuracy degrades by stacking an additional convolutional layer (refer to 3C-1S-FC SpiCNN in Fig. 5(c)). This is common in deep networks, where the optimal network depth depends on the target application. For MNIST digit recognition, we experimentally determined 2C-1S-FC SpiCNN (with 3x3 kernels and 16 feature maps per layer) as the optimal SpiCNN topology. Our analysis, shown in Fig. 5, further reveals that 1C-1S-FC SpiCNNs have improved the robustness for randomly initialized weights as the number of kernel increases across 5 simulation runs. On the other hand, networks with more hidden layers are comparatively sensitive to weight initialization with a standard deviation of 1.3%, 2.4%, and 3.8% respectively in classification accuracy for SpiCNN composed of one, two and three stacked convolutional layers.

Finally, we trained the 2C-1S-FC SpiCNN using grayscale images instead of binary images used for the previous analysis. We note that a grayscale image intrinsically encodes more information compared to its binary counterpart, which can be translated to the spiking domain effectively using the Poisson-distributed spike encoding scheme. However, our results indicated nearly 5% degradation in the classification accuracy using grayscale images relative to that achieved with binary images. We hypothesize that this could be an artifact of the input dataset, where the precise location of the edges in the MNIST data carries more significance than the absolute pixel
intensities at the respective locations. As mentioned earlier, all the edge pixels carry equal significance in the binary image irrespective of the absolute intensities (contained in the grayscale image), which is effectively translated to the spiking domain by using the same Poisson firing rate for all the edge pixels. Thus, the attributes of the input dataset determine which image type between binary and grayscale yields better classification accuracy. In order to validate our hypothesis, we demonstrate the utility of rich grayscale inputs and Poisson-distributed spike encoding scheme using natural real-world images in the subsequent analysis.

B. Caltech Image Recognition

We used a subset of 1226 images from the Caltech dataset [21] spanning two different object categories, namely, Face and Motorbike, for further illustrating the applicability of the proposed SpiCNN and the unsupervised convolutional STDP learning methodology. We randomly chose 200 images from each object category for training and the remaining for testing SpiCNN. Each individual Caltech image, originally encoded in high dimensional RGB colorspace, was LoG-filtered to obtain a single channel grayscale image with edges accentuated. The resultant grayscale image was resized and converted to a binary image of reduced dimension (28x36). We similarly use the binary images for determining the optimal topology, and finally highlight the benefits of directly training with the grayscale images for such real-world images.

We initially trained a 1C-1S-FC SpiCNN for different kernel sizes and number of kernels making up the convolutional layer. We observed that the feature representations acquired by certain kernels (for instance, the horizontal, vertical, and diagonal features in Fig. 8(a)) are similar to those learned from the MNIST digits (refer to Fig. 6(a)). This highlights the general feature learning capability of the presented convolutional STDP learning methodology. Our results (shown in Fig. 8(b)) indicate that the classification accuracy increases with the number of kernels with a kernel size of $7 \times 7$ yielding the highest accuracy of 95.6%. In an effort to achieve improved classification accuracy in a scalable manner, we explored the deep 2C-1S-FC and 3C-1S-FC SpiCNN configurations with 16 feature maps per layer. Our experimental analysis offered the following twin insights. First, the smaller $3 \times 3$ kernels performed better than the larger $5 \times 5$ and $7 \times 7$ kernels for deep SpiCNNs (as illustrated in Fig. 8(c)) owing to extract prominent features and construct distributed representation across successive hierarchical layers as shown in Fig. 7. Second, the 2C-1S-FC SpiCNN was found to be the optimal topology with a classification accuracy of 96.0% for the chosen Caltech image recognition task.

In our final experiment, we trained the optimal 2C-1S-FC SpiCNN directly using the grayscale images and achieved an improved accuracy of 97.6%. This is contrary to the trend we observed for MNIST dataset and essentially corroborates our hypothesis regarding the usefulness of the rich grayscale information for self-learning distinctive features from complex natural images. It is important to note that the chosen Poisson-distributed spike encoding scheme is capable of efficiently translating the grayscale information into spike trains, leading to improved feature learning by deep SpiCNN.

VI. DISCUSSION AND COMPARISON WITH RELATED WORKS

Our proposed deep SpiCNN achieves competitive classification accuracy across different datasets using fewer trainable parameters in comparison with typical fully-connected (shallow) SNNs. For instance, SpiCNN composed of two convolutional layers with 16 feature maps per layer provided a classification accuracy of 91.1% for MNIST digit recognition. On the other hand, the fully-connected SNN presented in [5] required 1600 excitatory (output) neurons amounting to 50x more trainable parameters to attain an equivalent accuracy. Furthermore, the fully-connected SNN was trained with a unit batch size to enable every excitatory neuron to learn a complete representation of a unique input pattern. Conversely, we used mini-batch learning that achieves feature learning with fewer synaptic weight updates and facilitates every convolutional kernel to self-learn features shared across different classes of input patterns. Furthermore, mini-batch learning help the network avoid drastic changes because of the individual training data that is far from generality. To these effect, the mini-batch learning renders convolutional kernels
TABLE III
CLASSIFICATION ACCURACY OF SNNs FOR CALTECH (FACE/MOTORBIKE).

<table>
<thead>
<tr>
<th>SNN Topology</th>
<th>Architecture</th>
<th>Spike Encoding Scheme</th>
<th>Learning Rule</th>
<th>#Trainable Parameters</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDNN [14]</td>
<td>Convolutional</td>
<td>Rank-order encoding</td>
<td>STDP + SVM</td>
<td>25480</td>
<td>99.1%</td>
</tr>
<tr>
<td>Spiking CNN [13]</td>
<td>Convolutional</td>
<td>Rank-order encoding</td>
<td>STDP + RBF</td>
<td>20480</td>
<td>97.7%</td>
</tr>
<tr>
<td>Spiking CNN [25]</td>
<td>Convolutional</td>
<td>Rank-order encoding</td>
<td>Reinforcement STDP</td>
<td>23720</td>
<td>98.9%</td>
</tr>
<tr>
<td>SpiCNN (our work)</td>
<td>Convolutional</td>
<td>Poisson-distributed spike encoding</td>
<td>STDP</td>
<td>25488</td>
<td>91.0%</td>
</tr>
</tbody>
</table>

TABLE IV
CLASSIFICATION ACCURACY OF SNNs FOR MNIST DIGIT RECOGNITION.

<table>
<thead>
<tr>
<th>SNN Topology</th>
<th>Architecture</th>
<th>Spike Encoding Scheme</th>
<th>Learning Rule</th>
<th>#Trainable Parameters</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two-layer SNN [5]</td>
<td>Fully-connected</td>
<td>Poisson-distributed spike encoding</td>
<td>STDP</td>
<td>5017600</td>
<td>95%</td>
</tr>
<tr>
<td>SDNN [14]</td>
<td>Convolutional</td>
<td>Rank-order encoding</td>
<td>STDP + SVM</td>
<td>76500</td>
<td>98.4%</td>
</tr>
<tr>
<td>Spiking CNN [27]</td>
<td>Convolutional</td>
<td>Poisson-distributed spike encoding</td>
<td>Sparse Coding + STDP + SVM</td>
<td>590642</td>
<td>98.3%</td>
</tr>
<tr>
<td>SpiCNN (our work)</td>
<td>Convolutional</td>
<td>Poisson-distributed spike encoding</td>
<td>STDP</td>
<td>25488</td>
<td>91.1%</td>
</tr>
</tbody>
</table>

TABLE V
CLASSIFICATION ACCURACY OF 2C-1S-FC SpiCNN ON THE MNIST TRAINING AND TESTING DATASET.

<table>
<thead>
<tr>
<th>Kernel Size</th>
<th>Training Accuracy</th>
<th>Testing Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>3×3</td>
<td>90.5%</td>
<td>91.1%</td>
</tr>
<tr>
<td>5×5</td>
<td>84.3%</td>
<td>85.0%</td>
</tr>
<tr>
<td>7×7</td>
<td>80.6%</td>
<td>82.4%</td>
</tr>
</tbody>
</table>

ANN-based classifier of Support Vector Machine or Radial Basis Function, respectively, to evaluate the effectiveness of extracted features. Our work differs from that presented in [13] and [14] in the following respects. First, we use Poisson-distributed spike encoding scheme that requires non-linear Leaky-Integrate-and-Fire (LIF) dynamics for better regulating the neuronal spiking activity. The chosen spike encoding scheme is implementation friendly since the individual spike events are uncorrelated, which requires the LIF neurons to simply integrate the weighted sum of incoming spikes. However, the rank order spike encoding scheme necessitates hardware overhead to account for the relative order of spikes emitted by the individual neurons. This is illustrated in [11], which shows a feed-forward SNN with inhibitory inter-neurons that reduce the effectiveness of the input spikes feeding the excitatory neurons based on the respective order of firing. Additionally, temporal rank-order spike encoding scheme is sensitive to variability in spike timings caused by intrinsic neuronal noise typical in custom hardware realizations using analog-CMOS or emerging technologies [15]. Second, we demonstrate the ability of smaller 3×3 kernels in all the convolutional layers containing the same number of feature maps to learn generalized prominent features as a result of having fewer trainable parameters. Last, we perform mini-batch learning that leads to general feature learning using fewer synaptic weight updates. Table III shows that SpiCNN offers comparable classification accuracy for Caltech (Face/Motorbike) image recognition. However, the classification accuracy of SpiCNN is lower than that reported in mbox [14] and [27] for MNIST digit recognition as shown in Table IV. However, related works [14], [27] used ANN-based readout for final classification upon training the convolutional layers with STDP while we train all the layers of SpiCNN including the final classification layer using greedy layer-wise STDP through direct spike events. It is important to note that SpiCNN offers competitive classification accuracy with fewer trainable parameters as illustrated in Table IV. Future works could explore enhanced learning mechanisms like reward-modulated STDP to further improve the performance of multi-layer SNNs without using external classifiers as demonstrated in [26], [32], [33].

to efficiently learn better features that contain the generalized characteristics. Hence, SpiCNN yields competitive accuracy with fewer synaptic weight updates.

Deep SNNs varying in degrees of bio-fidelity have been proposed in the literature. In [28] and [29], a deep SNN is proposed which is trained offline using the backpropagation algorithm as an Artificial Neural Network (ANN). The ANN-to-SNN conversion merely exploits the event-driven processing capability of SNNs for achieving energy efficiency during inference while trading off the on-chip learning capability. Furthermore, mapping the trained weights of an ANN to the corresponding SNN leads to a loss in the classification accuracy. Researchers in [22], [30], [31] directly trained deep SNNs through spike events using neuronal spiking rate-based and membrane potential-based error backpropagation algorithm, respectively. Nevertheless, the computational complexity incurred during training hinders on-chip implementation. Our deep SpiCNN is trained using bio-inspired STDP-based unsupervised learning that harnesses the event-driven processing and on-chip learning capabilities. Furthermore, as depicted in Table V, estimated training accuracies of 2C-1S-FC SpiCNN are not higher than testing accuracies on handwritten digit (MNIST) datasets. SpiCNNs trained by greedy layer-wise unsupervised STDP learning are less subjective to an overfitting phenomenon, which is commonly observed in supervised backpropagation algorithm.

Recent works have explored the use of STDP for training multi-layer SNNs. In [13], an illustration of the applicability of STDP-based visual feature learning on SNNs with single convolutional layer is shown while reference [14] successfully trained SNNs with multiple convolutional layers. Both [13] and [14] used temporal rank-order spike encoding scheme in a network of integrate-and-fire neurons and trained convolutional layers for input feature extraction using STDP and
VII. Conclusion

The application of STDP-based unsupervised learning has thus far been primarily limited to shallow fully-connected SNN topologies, which necessitates large number of trainable parameters to achieve competitive classification accuracy. In this work, we proposed a deep SpiCNN composed of a hierarchy of stacked convolutional layers followed by a spatial-pooling layer and a fully-connected layer for self-learning input features using fewer trainable parameters. We hierarchically trained the shared synaptic weight kernels interconnecting successive convolutional layers using STDP for unsupervised feature extraction. Furthermore, we demonstrated improved feature learning using smaller $3 \times 3$ kernels trained with STDP-based mini-batch synaptic weight updates. We validated the efficacy of the presented unsupervised convolutional STDP learning methodology by training deep SpiCNN to effectively recognize handwritten MNIST digits and natural Caltech images. The reduction in the number of trainable parameters (for smaller $3 \times 3$ kernels) and the frequency of synaptic weight updates (as a result of mini-batch training) coupled with the use of robust Poisson-distributed spike encoding scheme (for layer-wise training) render SpiCNN amenable for energy-efficient neuromorphic hardware implementations.

REFERENCES


Chankyu Lee received B.S. in Electrical and Electronics Engineering from Sungkyunkwan University, Korea, in 2015. Currently, he is pursuing PhD degree in Electrical and Computer Engineering at Purdue University, West Lafayette, IN, USA. His primary research lies in the area of brain-inspired (neuromorphic) computing and event-driven deep learning, low power and high performance VLSI design for machine learning hardware.

Gopalakrishnan Srinivasan is currently pursuing his PhD in Electrical Engineering at Purdue University under the guidance of Prof. Kaushik Roy. His primary research interests include investigating bio-inspired spiking neuromorphic computing paradigms and their energy-efficient implementation using CMOS and post-CMOS (spintronic) technologies. He received his B.Tech. in Electrical and Electronics Engineering from the National Institute of Technology, Calicut, India, and his Masters in Computer Engineering from the North Carolina State University, Raleigh, NC, in 2010 and 2012 respectively.

Priyadarshini Panda 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 interests include low-power neuromorphic computing: energy-efficient realization of neural networks (spiking/non-spiking in deep learning context) using novel architectures and algorithms.
Kaushik Roy received B.Tech. degree in electronics and electrical communications engineering from the Indian Institute of Technology, Kharagpur, India, and Ph.D. degree from the electrical and computer engineering department of the University of Illinois at Urbana-Champaign in 1990. He was with the Semiconductor Process and Design Center of Texas Instruments, Dallas, where he worked on FPGA architecture development and low-power circuit design. He joined the electrical and computer engineering faculty at Purdue University, West Lafayette, IN, in 1993, where he is currently Edward G. Tiedemann Jr. Distinguished Professor. He also the director of the center for brain-inspired computing (C-BRIC) funded by SRC/DARPA. His research interests include neuromorphic and emerging computing models, neuro-mimetic devices, spintronics, device-circuit-algorithm co-design for nano-scale Silicon and non-Silicon technologies, and low-power electronics. Dr. Roy has published more than 700 papers in refereed journals and conferences, holds 18 patents, supervised 75 PhD dissertations, and is co-author of two books on Low Power CMOS VLSI Design (John Wiley & McGraw Hill).