RMP-SNN: Residual Membrane Potential Neuron for Enabling Deeper High-Accuracy and Low-Latency Spiking Neural Network

Bing Han, Gopalakrishnan Srinivasan, and Kaushik Roy
School of Electrical and Computer Engineering, Purdue University
{han183,srinivg,kaushik}@purdue.edu

Abstract

Spiking Neural Networks (SNNs) have recently attracted significant research interest as the third generation of artificial neural networks that can enable low-power event-driven data analytics. The best performing SNNs for image recognition tasks are obtained by converting a trained Analog Neural Network (ANN), consisting of Rectified Linear Units (ReLU), to SNN composed of integrate-and-fire neurons with "proper" firing thresholds. The converted SNNs typically incur loss in accuracy compared to that provided by the original ANN and require sizable number of inference time-steps to achieve the best accuracy. We find that performance degradation in the converted SNN stems from using "hard reset" spiking neuron that is driven to fixed reset potential once its membrane potential exceeds the firing threshold, leading to information loss during SNN inference. We propose ANN-SNN conversion using "soft reset" spiking neuron model, referred to as Residual Membrane Potential (RMP) spiking neuron, which retains the "residual" membrane potential above threshold at the firing instants. We demonstrate near loss-less ANN-SNN conversion using RMP neurons for VGG-16, ResNet-20, and ResNet-34 SNNs on challenging datasets including CIFAR-10 (93.63% top-1), CIFAR-100 (70.93% top-1), and ImageNet (73.09% top-1 accuracy). Our results also show that RMP-SNN surpasses the best inference accuracy provided by the converted SNN with "hard reset" spiking neurons using 2-8× fewer inference time-steps across network architectures and datasets.

1. Introduction

Deep neural networks, referred to as Analog Neural Networks (ANNs) in this article, composed of several layers of interconnected neurons, have achieved state-of-the-art performance in various Artificial Intelligence (AI) tasks including image localization and recognition [18, 31], video analytics [28], and natural language processing [16], among other tasks. The superior performance has been achieved by trading off computational efficiency. For instance, ResNet [11] that won the ImageNet Large Scale Visual Recognition Challenge in 2015 consists of 152 layers with over 60 million parameters, and incurs 11.3 billion FLOPS per classification. In an effort to explore more power efficient neural architectures, recent research efforts have been directed towards devising computing models inspired from biological neurons that compute and communicate using spikes. These emerging class of networks with increased bio-fidelity are known as Spiking Neural Networks (SNNs)[22]. The intrinsic power-efficiency of SNNs stems from their sparse spike-based computation and communication capability, which can be exploited to achieve higher computational efficiency in specialized neuromorphic hardware [2, 4, 24].

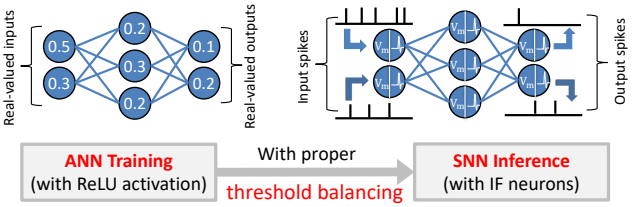

Figure 1. Illustration of the ANN-SNN conversion methodology.

Considering the rapid strides in accuracy achieved by ANNs over the past few years, SNN training algorithms are much less mature and are an active field of research. The training algorithms for SNNs can be categorized into Spike Timing Dependent Plasticity (STDP) based localized learning rules, spike-based error backpropagation, and ANN-SNN conversion methodologies. STDP-based unsupervised [5, 23, 37, 40] and semi-supervised learning algorithms [17, 20, 25, 39] have thus far been restricted to shallow SNNs (with $\leq$5 layers) yielding considerably lower accuracy than that provided by ANNs on complex datasets like CIFAR-10 [9, 38]. In order to scale the networks much deeper, spike-based error backpropagation algorithms have been proposed for the supervised training of SNNs [1, 15, 19, 21, 27, 29, 35, 41]. The training complexity incurred for performing error backpropagation over time has limited their scalability for SNNs beyond 9-11 layers [19].

ANN-SNN conversion has yielded the best performing SNNs (typically composed of Integrate-and-Fire (IF) neurons), which are converted from a trained non-spiking ANN (consisting of Rectified Linear Unit (ReLU) as the activation function) [3, 6, 7, 30, 34, 42] as illustrated in Fig. 1. The conversion schemes intelligently assign "appropriate" firing thresholds to the neurons at different layers of the network, thereby, ensuring that the IF spiking rates (number of spikes over large enough time interval) are proportional to the corresponding analog ReLU activations. Such conversion approaches take full advantage of backpropagation-based training, well-developed for ANNs. Note, however, the converted SNNs do not have the accuracy of the corresponding ANNs and require a sizeable number of time-steps (>2000 for ImageNet [34]) for achieving the best accuracy (69.96% [34]). We find that performance degradation in the converted network stems from using spiking IF neurons, with "hard reset" mechanism, to map analog activations to spike rates. A "hard reset" neuron is driven to *a priori* fixed low potential once its internal state (or membrane potential) exceeds the firing threshold, irrespective of how high the membrane potential is above the threshold. We find that ignoring the "residual" potential above threshold leads to information loss in the conversion process (from ReLU-based artificial neurons to "hard reset" IF neurons).

We propose conversion-based training using "soft reset" spiking neuron, referred to as Residual Membrane Potential (RMP) spiking neuron, which better mimics the ReLU functionality. The RMP neuron keeps the "residual" potential above firing threshold at the spiking instants instead of "hard reset" to fixed potential, thereby alleviating the information loss that occurs during ANN-SNN conversion. We implemented deep SNN architectures such as VGG-16 and residual networks (ResNet-20 and ResNet-34) using RMP spiking neurons and demonstrate near loss-less conversion with close to state-of-the-art accuracy on complex datasets including ImageNet. We note that RMP neurons have been used for realizing deep SNNs that are converted from non-spiking ANNs [32, 33], albeit with higher conversion loss during SNN inference. We present the appropriate threshold initialization scheme to achieve near loss-less mapping of ReLU activations to RMP neuron spiking rates, yielding SNNs that provide the best inference accuracy to date on CIFAR-10, CIFAR-100, and ImageNet datasets. In addition, we demonstrate the ability of RMP-SNN to offer competitive accuracy using up to 8$\times$ fewer inference time-steps compared to converted SNN with "hard reset" neurons, with only 1-2% increase in overall spiking activity.

2. Related Work

ANN-SNN conversion has been shown to be promising approach for building deep SNNs yielding high enough accuracy for complex image recognition [3, 6, 30, 32, 33, 34, 42] and natural language processing tasks [7]. The conversion schemes train ANN, composed of ReLU non-linearity, using backpropagation with added constraints like removal of bias neurons and batch normalization layers. The trained ANN is mapped to SNN composed of IF neurons. A notable exception to ReLU-IF mapping is the work of Hunsberger et al. [13] who used more bio-plausible Leaky-Integrate-and-Fire (LIF) neuron during inference by training the ANN with rate-based soft-LIF non-linearity. Efficient ANN-SNN conversion requires careful initialization of thresholds at every layer of the network so that the spiking rates are proportional to the ReLU activations. In this regard, Deihl et al. [6] proposed model-based and data-based threshold balancing schemes. Model-based scheme estimates the threshold using only the ANN weights while the data-based scheme uses both the training data and weights. Following the work of [6], Sengupta et al. [34] proposed data-based scheme that additionally uses SNN spiking statistics to achieve much improved ANN-SNN conversion, which has been shown to scale well to complex datasets like ImageNet. However, the aforementioned approaches are inherently susceptible to information loss during inference due to the use of "hard reset" neurons as will be explained in section 3.2. Rueckauer et al. [33] attempted to mitigate the information loss by using "soft reset" RMP neurons. However, they report significantly high accuracy loss (14.28%) between VGG-16 ANN and SNN on ImageNet compared to that (1.13%) incurred by the approach of Sengupta et al. [34] using "hard reset" neurons. We believe that the higher accuracy loss incurred by Rueckauer et al. [33] is a consequence of unconstrained training of ANN with bias neurons and batch normalization layers as also pointed out in [34]. While removing the constraints improved the accuracy of ANN, the converted SNN suffered from substantial accuracy loss, thereby, hiding the potential benefits of RMP neurons. We perform constrained ANN training and demonstrate near loss-less low-latency ANN-SNN conversion with "soft reset" RMP neurons. The novelty of our work is the proposal of ANN-SNN conversion methodology using a combination of "soft reset" RMP spiking neuron, appropriate layer-wise threshold initialization, and constrained ANN training (removal of batch normalization layers and bias neurons) to enable near loss-less ANN-SNN conversion.

3. ANN-SNN Conversion

The fundamental distinction between ANN and SNN is the notion of time. In ANNs, input and output of neurons in all the layers are real-valued, and inference is performed with single feed-forward pass through the network. On the other hand, input and output of spiking neurons are encoded temporally using sparse spiking events over certain time period. Hence, inference in SNNs is carried out over multiple feed-forward passes or time-steps (also known as inference

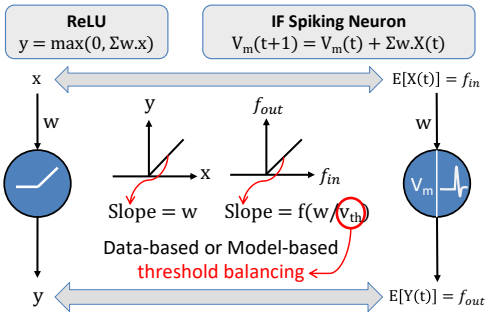

Figure 2. Illustration of ReLU-IF mapping. The IF neuron threshold is set using model-based [6] or data-based schemes [6, 34] so that its output rate is proportional to the ReLU activation.

latency), where each pass entails sparse spike-based computations. Achieving close to ANN accuracy with minimal inference latency is key to obtaining favorable trade-off between accuracy and computational efficiency. The proposed conversion methodology significantly advances the state-of-the-art in this regard as will be detailed in section 5.

3.1. Input Encoding for SNNs

We use Poisson rate coding to map the input image pixels to spike trains firing at a rate (number of spikes over time) proportional to the corresponding pixel intensities as shown in [12]. The pixel intensity is first mapped to instantaneous spiking probability of the corresponding input neuron. We use Poisson process to generate the input spike in a stochastic manner as explained below. At every time-step of SNN operation, we generate a uniform random number between 0 and 1, which is compared against the neuronal firing probability. A spike is produced if the random number is less than the neuronal firing probability. Note that, input images fed to ANN are typically normalized to zero mean and unit standard deviation, yielding pixel intensities between ± 1. For the SNN, we generate positive or negative spikes based on the sign of the normalized intensities. The time interval (inference latency) is dictated by the desired accuracy.

3.2. ReLU-IF Mapping with Hard Reset

ANNs used for conversion to SNNs are typically trained with ReLU non-linearity [26], which is described by

$$Y = max\,(0, X) \tag{1}$$

where Y is the output of ReLU-based artificial neruon, $X = \sum_i w_i.x_i + b$ is the weighted sum of input x_i with weight w_i and bias b. The bias is usually set to zero for effective ANN-SNN conversion [34]. The ReLU ouput varies linearly with positive inputs. The linear ReLU dynamics are roughly mimicked using Integrate-and-Fire (IF) neuron as illustrated in Fig. 2. An IF neuron receives train of spikes, over certain time period, whose rate corresponds to the real-valued ReLU input. The IF neuron integrates the weighted

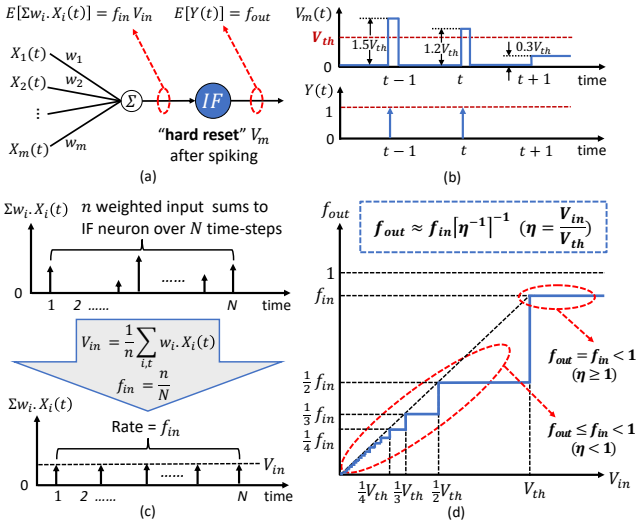

Figure 3. (a) "Hard reset" IF neuron driven by a set of input neurons via weights (w). (b) Illustration of reduced firing rate of IF neuron due to resetting the membrane potential (V_m) to zero at the spiking instants. (c) Mapping the expectation of weighted input sum received by the IF neuron over time to the product of average rate f_{in} and amplitude V_{in}. (d) Non-linear input-output (f_{in}-f_{out}) response of IF neuron for different V_{in}.

spike-input into its membrane potential whose dynamics are described by

$$V_m(t) = V_m(t-1) + \sum_i w_i.X_i(t) \tag{2}$$

where $V_m(t)$ is the membrane potential at time-step t, w_i is the transferred weight from ANN, and X_i is the spike train of i-th input neuron. The IF neuron produces a spike when its membrane potential exceeds the firing threshold V_{th} (> 0), which is estimated using model-based [6] or data-based schemes [6, 34]. At the instant of a spike, the membrane potential is "hard reset" to 0 irrespective of the amount by which the membrane potential exceeds the threshold. Ignoring the residual membrane potential above threshold affects the expected linear relationship between the input and the output spiking rates as illustrated below with an example. Let us suppose that an IF neuron (shown in Fig. 3(a)) receives weighted input sum of $1.5V_{th}$, $1.2V_{th}$, and $0.3V_{th}$ in three successive time-steps as depicted in Fig. 3(b). The total weighted input sum across the three time-steps is $3V_{th}$. The IF neuron needs to fire thrice to maintain precise linear relationship between the input and output spiking rates. In effect, it generates only two spikes over three time-steps as a result of ignoring the residual potential above threshold at the firing instants as illustrated in Fig. 3(b).

We now formalize the deviation of the input-output relationship of "hard reset" neuron from the expected linear behavior. The average weighted input sum, $E[\sum_i w_i.X_i(t)]$,

received by the IF neuron can be specified as $f_{in}\,V_{in}$, where f_{in} and V_{in} are the mean rate and amplitude of the weighted input sum, respectively, as shown in Fig. 3(c). The average input amplitude, V_{in}, can moreover be specified as $\eta\,V_{th}$ for $\eta \in \mathbb{R}^+$ without any loss of generality. The output firing rate (number of output spikes over time), f_{out}, of the IF neuron is then described by

$$f_{out} = \begin{cases} f_{in} & \eta = \frac{V_{in}}{V_{th}} \geq 1 \\ \lfloor f_{in} \lceil \eta^{-1} \rceil^{-1} N \rfloor N^{-1} & 0 \leq \eta < 1 \\ \approx f_{in} \lceil \eta^{-1} \rceil^{-1} & 0 \leq \eta < 1 \; and \; N \gg 1 \end{cases} \tag{3}$$

in which $\lceil\ \rceil$ is the ceiling operation, $\lfloor\ \rfloor$ is the floor operation, $f_{out} \leq f_{in} \leq 1$, and N is the total number inference time-steps. The output rate matches the input rate only for $\eta \geq 1$ when the average input amplitude V_{in} is larger than the threshold V_{th}. In this case, the input amplitude is high enough to warrant an output spike every time it occurs in spite of ignoring the residual potential from earlier spiking instants. On the other hand, when $0 \leq \eta < 1$, the output spiking rate f_{out} is approximately specified by $\lceil \eta^{-1} \rceil^{-1} f_{in}$ (for large enough N) as described in equation 3. The ceiling operation accounts for the non-linear relationship between the input and output rates as illustrated in Fig. 3(d) and explained below. Consider $\eta \in [\frac{1}{k+1}, \frac{1}{k})$ and $V_{in} \in [\frac{V_{th}}{k+1}, \frac{V_{th}}{k})$ for any positive integer k. As the average input amplitude, V_{in}, gradually changes from $\frac{V_{th}}{k+1}$ to $\frac{V_{th}}{k}$, the output rate f_{out} remains constant at $\frac{f_{in}}{k+1}$ instead of linearly increasing to $\frac{f_{in}}{k}$ as shown in Fig. 3(d). For instance, let us suppose that $\eta \in [\frac{1}{2}, 1)$ and $V_{in} \in [\frac{V_{th}}{2}, V_{th})$ while f_{in} is unity. If V_{in} of $\frac{V_{th}}{2}$ is received per time-step, the IF neuron fires a spike every second time-step. When V_{in} increases to $\frac{3\,V_{th}}{4}$ per time-step, the IF neuron still fires a spike only every second time-step instead of firing 3 spikes over 4 time-steps. The reduced output rate is a direct consequence of ignoring the residual potential at the firing instants by performing "hard reset" to 0.

Fig. 3(d) shows that roughly linear input-output relationship can be obtained for $\eta \ll 1$, which requires the average input amplitude to be much lower than the firing threshold of "hard reset" IF neurons. Low input activity in converted SNN can be achieved by setting the layer-wise thresholds to be much higher, which reduces the ANN-SNN conversion loss at the cost of significantly high inference latency. On the other hand, lowering the thresholds, to minimize the inference latency, increases the layer-wise input spiking activity, resulting in η closer to 1. Higher η causes the "hard reset" neurons to operate in the non-linear regime, leading to larger degradation in SNN accuracy. ANN-SNN conversion with "hard reset" neurons requires careful initialization of thresholds to obtain favorable accuracy-latency trade-off.

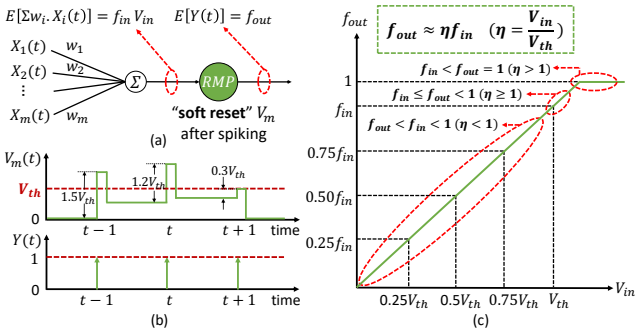

Figure 4. (a) "Soft reset" RMP neuron driven by a set of input neurons via weights (w). (b) Illustration of precise spiking behavior of RMP neuron by retaining the residual potential at the firing instants. (c) Linear input-output (f_{in}-f_{out}) response of RMP neuron for different V_{in}.

4. Residual Membrane Potential SNN

We propose Residual Membrane Potential (RMP) spiking neuron (shown in Fig. 4(a)) to obtain linear input-output characteristics and achieve near loss-less ANN-SNN conversion. The RMP neuron minimizes information loss during inference by performing "soft reset" as described in the following pseudo-code.

```
if V_m(t) ≥ V_th :
    Emit Output Spike: Y(t) = 1
    Perform Soft Reset: V_m(t) = V_m(t) − V_th
```

At the instant of a spike, the membrane potential (V_m) is reduced by an amount equal to the firing threshold (V_{th}) instead of "hard reset" to 0. "Soft reset" effectively retains the residual potential above threshold as shown in Fig. 4(b). Let us suppose that an RMP neuron receives weighted input sum of $1.5V_{th}$, $1.2V_{th}$, and $0.3V_{th}$, totalling to $3V_{th}$, across three consecutive time-steps. It produces the expected number of three spikes by retaining the residual potential at the firing instants as depicted in Fig. 4(b). Note that "soft reset" is also referred to as "reset by subtraction" in SNN literature [32, 33]. Formally, the output firing rate, f_{out}, of RMP neuron can be described by

$$f_{out} = \begin{cases} \lfloor \eta f_{in} N \rfloor N^{-1} & \eta \geq 0 \\ \approx \eta f_{in} & \eta \geq 0 \; and \; N \gg 1 \end{cases} \tag{4}$$

in which $f_{in} \leq 1$, $f_{out} \leq 1$, $\eta = \frac{V_{in}}{V_{th}}$ is the ratio between the average amplitude of weighted input sum V_{in} and firing threshold V_{th}, and N is the inference latency. The output rate changes proportional to the input rate by a factor η for a wide range of η as depicted in Fig. 4(c) due to carrying over the residual potential at the firing instant to the following time-step. The linear input-output characteristics exhibited by the RMP neuron for a wide range of η

enable it to provide near loss-less ANN-SNN mapping for a wide range of firing thresholds as will be discussed in the following section 4.1.

4.1. Threshold Balancing for RMP-SNN

ANN-SNN conversion requires assigning "appropriate" threshold for the spiking neurons to ensure that they operate in the linear (or almost linear) regime, which effectively leads to lower (or even negligible) conversion loss. The extended linear input-output relationship of RMP neuron (see Fig. 4(c)) provides "wider operating range" for the neuronal firing threshold compared to that for the "hard reset" IF neuron (refer to Fig. 3(d)). This begets the following couple of questions that need to be answered to ensure appropriate threshold balancing for the RMP neuron.

1. For any given f_{in} and V_{in}, what is the desired operating range for the RMP neuron firing threshold to ensure loss-less ANN-SNN conversion?

2. How should the absolute value of threshold be determined so that the RMP neuron operates in the desired range?

We determine the upper and lower bounds for the RMP neuron firing threshold based on the desired operating range for the output rate f_{out}. Fig. 4(c) indicates that the RMP neuron allows f_{out} to be higher than the input rate f_{in}, while still exhibiting linear input-output dynamics. The desirable range for f_{out} is $[f_{in}, 1)$. This is because $f_{out} \geq f_{in}$ ensures sufficient spiking activity across successive layers of deep SNN, leading to high enough accuracy using fewer inference time-steps. Satisfying $f_{out} \geq f_{in}$ requires $\eta = \frac{V_{in}}{V_{th}} \geq 1$ or $V_{th} \leq V_{in}$ as highlighted in Fig. 4(c). On the other hand, $f_{out} < f_{in}$ (or $V_{th} > V_{in}$) leads to gradual reduction in SNN spiking activity with network depth, thereby, increasing the inference latency.

Next, the lower bound for V_{th} is determined to ensure that the output rate f_{out} is less than unity. This is because, $f_{out} \geq 1$ produces a spike at every time-step irrespective the received input. The excessive spiking activity can lead to substantial degradation in accuracy. The threshold required for guaranteeing $f_{out} < 1$ can be obtained from the following equation that relates the average input and output potentials, which is described by

$$\frac{dV_m^{avg}}{dt} = f_{in}V_{in} - f_{out}V_{th} \tag{5}$$

where V_m^{avg} is the average membrane potential of the RMP neuron. In order for the average potential to reach its steady state value, $\frac{dV_m^{avg}}{dt}$ in (5) must be equal to zero. The output rate f_{out} is then specified by the following equation.

$$f_{out} = \frac{f_{in}V_{in}}{V_{th}} \tag{6}$$

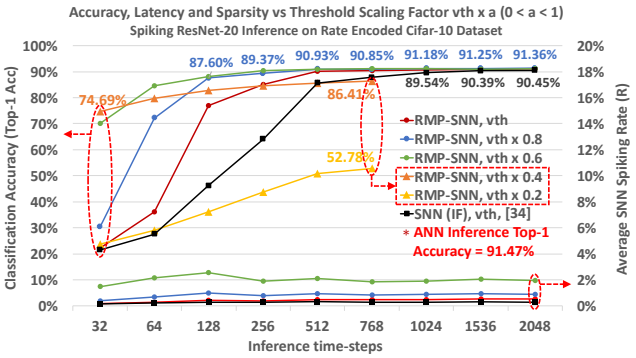

Figure 5. Inference accuracy and spiking activity versus latency of ResNet-20 SNN, composed of RMP neurons, for different threshold scaling factor α on the CIFAR-10 dataset.

Equation 6 clearly indicates that V_{th} must be greater than $f_{in}V_{in}$ for f_{out} to be smaller than 1. Thus, the desired operating range of V_{th}, for given f_{in} and V_{in}, is specified by

$$f_{in}V_{in} \leq V_{th} \leq V_{in} \tag{7}$$

which answers the first question raised in the beginning of this section. As explained previously, the V_{th} range specified in (7) ensures $f_{in} \leq f_{out} < 1$, which can lead to optimal accuracy-latency trade-off. It is important to note that f_{out} for "hard reset" IF neuron is always smaller than or equal to input rate f_{in} (as shown in Fig. 3(d)) due to ignoring the residual potential. As a result, the "hard reset" IF neuron inherently incurs higher inference latency compared to that for the RMP neuron.

We now address the second question concerning the precise V_{th} estimation methodology. In our analysis thus far, we estimated V_{th} using $V_{in} = E[\sum_i w_i X_i(t)]$, which is the average weighted input sum to the RMP neuron over time. Prior works proposed setting V_{th} to $Max_t[\sum_i w_i X_i(t)]$, which is the maximum weighted input sum to the neuron across time-steps [6, 34]. The maximum estimate V_{in}^{max} can enable the RMP neuron to operate in the linear region (where $f_{out} = f_{in}$ as highlighted in Fig. 4(c)), while the average estimate V_{in}^{avg} ($< V_{in}^{max}$) can cause it to operate more in the vicinity of the non-linear region (where $f_{out} = 1$ designated in Fig. 4(c)). We validate our hypothesis of using V_{in}^{max} versus V_{in}^{avg} ($\approx \alpha V_{in}^{max}$ where $\alpha \in (0, 1)$ is a scaling factor) using ResNet-20 SNN on the CIFAR-10 dataset. Before presenting the results, we describe the methodology, originally proposed in [34], used to initialize the layer-wise threshold of deep SNN using the ANN-trained weights and SNN spiking statistics. We transfer the trained weights from ANN to SNN, and feed the Poisson spike-inputs (for the entire training set) to the first layer of the SNN. We record the weighted input sum to all the neurons in the first layer across time-steps. We set the threshold of RMP neurons in the first layer to the maximum weighted input sum, across neurons

and time-steps, over the training dataset. We then freeze the threshold of the first layer, and estimate the threshold of the second layer using the same procedure outlined previously. The threshold estimation process is carried out sequentially in a layer-wise manner for all the layers.

ResNet-20 SNN, with its layer-wise threshold assigned to V_{in}^{max}, achieved 91.36% on CIFAR-10, which is comparable to that (91.47%) achieved by the corresponding ANN as illustrated in Fig. 5. We thereafter scaled the threshold by a factor of up to 0.6× and found that the RMP-SNN, with scaled threshold, converged to the same accuracy obtained using V_{th}=V_{in}^{max}. This corroborates our hypothesis that the RMP neuron operates in the linear region for a wide range of firing thresholds, thereby, causing the RMP-SNN to yield higher accuracy using fewer time-steps as depicted in Fig. 5. As the threshold is scaled further by up to 0.2×, we notice significant drop in accuracy. At such low thresholds, the RMP neuron operates in the non-linear (excessive spiking) regime, leading to higher accuracy loss during inference. We propose initializing the threshold of RMP-SNN with scaled version of V_{in}^{max} (scaling factor $\alpha \leq 0.6$ in this example) to achieve the optimal accuracy-latency trade-off. We validate the presented threshold initialization scheme across different SNN architectures and datasets.

Improving the inference latency by reducing the firing threshold increases the spiking activity, thereby, adversely impacting the overall computational efficiency. In an effort to quantify the spiking activity of RMP-SNN for different thresholds, we measure the average spike rate as defined by the following equation.

$$R = \frac{total\ spikes}{total\ neurons \times\ inference\ time\text{-}steps} \times 100\% \tag{8}$$

The spike rate R in (8) indicates the average percentage of neurons that spike per time-step. Our analysis indicates that the RMP-SNN, with scaled thresholds, provides disproportionate benefits in accuracy and latency compared to the increase in spiking activity (∼1-2%) as will be discussed in section 5.

5. Results

We evaluated RMP-SNNs on standard visual object recognition benchmarks, namely the CIFAR-10, CIFAR-100 and ImageNet datasets. We use VGG-16 architecture [36] for all three datasets. ResNet-20 configuration outlined in [11] is used for the CIFAR-10 and CIFAR-100 datasets while ResNet-34 is used for experiments on the ImageNet dataset. Our implementation is derived from the Facebook ResNet implementation code for CIFAR and ImageNet datasets. The code can be found online at https://github.com/facebookarchive/fb.resnet.torch. Proper weight initialization is crucial to achieve convergence in such deep networks without batch-normalization. Similar weights initialization was done as outlined in [10] although their networks were trained without both dropout and batch-normalization. For VGG networks, a dropout layer is used after every ReLU layer except for those layers which are followed by a pooling layer. For Residual networks, we use dropout only for the ReLUs at the non-identity parallel paths but not at the junction layers. We found this to be crucial for achieving training convergence.

The most recent state-of-the-art ANN-SNN conversion works are provided for comparison as shown in Table.1, 2 and 3. Note that authors in [33] reported a top-1 SNN error rate of 25.04% for an Inception-V3 network, with their ANN trained to an error rate of 23.88%. The resulting conversion loss is 1.52% which is much higher than our proposal. The Inception-V3 network conversion was also optimised by a voltage clamping method, that was found to be specific for the Inception network and did not apply to the VGG network [33]. In addition, the results reported on ImageNet in [33] are on a subset of 1382 image samples for Inception-V3 network and 2570 samples for VGG-16 network. Hence, the performance on the entire dataset is unclear. Our proposed RMP-SNN achieved not only the best SNN inference accuracies but also the lowest ANN-SNN conversion loss across all network architectures and datasets we evaluated. All SNN results reported represent the average of 5 independent runs. RMP-SNNs performances on accuracy, latency and sparsity are also presented and compared with the best performing SNNs to date in [34] as shown in Fig.5 to Fig.10. In each figure, x-axis is the SNN inference latency, the y-axis on the left measures the SNN top-1 inference accuracy, and the y-axis on the right measures the average spike rate.

The VGG-16 RMP-SNN inference on CIFAR-10 dataset is shown in Fig.6. RMP-SNN achieved the same accuracy 93.63% as the trained ANN using 2048 time-steps, whereas the SNN with IF neurons achieved 93.50% at the end of 2048 time-steps. The fastest RMP-SNN with reduced threshold (green curve) reaches an accuracy above 90% using only 64 time-steps, which is 8 times faster than the baseline SNN with IF neurons that uses 512 time-steps. Reducing thresholds causes an increase in spike rate. However, the fastest RMP-SNN with reduced threshold (green curve) still attains a spike rate less than 2%. Note, in this work, we reported higher accuracy of the baseline SNN with IF neurons compared to [34], in which, their best accuracy of the VGG-16 SNN with IF neurons on CIFAR-10 dataset is 91.55%. This is because we trained the ANN to a higher accuracy than the one used in [34] and the baseline SNN with IF neurons in our work is converted from the better trained ANN.

The VGG-16 RMP-SNN inference on CIFAR-100

Table 1. Accuracy loss due to ANN-SNN conversion of the state-of-the-art SNNs on CIFAR-10 dataset

Network Architecture	Spiking Neuron Model	ANN (Top-1 Acc)	SNN (Top-1 Acc)	Accuracy Loss
8-layered [14]	LIF (hard-reset)	83.72%	83.54%	0.18%
3-layered [8]	LIF (hard-reset)	-	89.32%	-
6-layered [33]	IF (hard-reset)	91.91%	90.85%	1.06%
ResNet-20 [34]	IF (hard-reset)	89.1%	87.46%	1.64%
ResNet-20 [This Work]	**RMP (soft-reset)**	**91.47%**	**91.36%**	**0.11%**
VGG-16 [34]	IF (hard-reset)	91.7%	91.55%	0.15%
VGG-16 [This Work]	**RMP (soft-reset)**	**93.63%**	**93.63%**	**$<$ 0.01%**

Table 2. Accuracy loss due to ANN-SNN conversion of the state-of-the-art SNNs on CIFAR-100 dataset

Network Architecture	Spiking Neuron Model	ANN (Top-1 Acc)	SNN (Top-1 Acc)	Accuracy Loss
ResNet-20 [34]	IF (hard-reset)	68.72%	64.09%	4.63%
ResNet-20 [This Work]	**RMP (soft-reset)**	**68.72%**	**67.82%**	**0.9%**
VGG-16 [34]	IF (hard-reset)	71.22%	70.77%	0.45%
VGG-16 [This Work]	**RMP (soft-reset)**	**71.22%**	**70.93%**	**0.29%**

Table 3. Accuracy loss due to ANN-SNN conversion of the state-of-the-art SNNs on ImageNet dataset

Network Architecture	Spiking Neuron Model	ANN (Top-1 Acc)	SNN (Top-1 Acc)	Accuracy Loss
ResNet-34 [34]	IF (hard-reset)	70.69%	65.47%	5.22%
ResNet-34 [This Work]	**RMP (soft-reset)**	**70.64%**	**69.89%**	**0.75%**
VGG-16 [33]	RMP (soft-reset)	63.89%	49.61%	14.28%
VGG-16 [34]	IF (hard-reset)	70.52%	69.96%	0.56%
VGG-16 [This Work]	**RMP (soft-reset)**	**73.49%**	**73.09%**	**0.4%**

dataset is shown in Fig.7, which reaches an accuracy of 70.93% using 2048 time-steps, whereas the baseline SNN with IF neurons reaches 70.77% at the end of 2048 time-steps. Note, no VGG-16 SNN was evaluated on CIFAR-100 dataset in [34]. In this work, both RMP-SNN and the baseline SNN with IF neurons were converted from our trained ANN with top-1 inference accuracy of 71.22%. The RMP-SNN with reduced threshold (blue curve) reaches an accuracy of 68.34% using only 256 time-steps, which is 2 times faster than the baseline SNN with IF neurons that uses about 512 time-steps. The RMP-SNN with reduced threshold (blue curve) attains a spike rate lower than 1% throughout the entire inference time-steps.

The VGG-16 RMP-SNN inference on the ImageNet dataset is shown in Fig.8. It reaches an accuracy of 73.09% using 4096 time-steps, whereas the SNN with IF neurons reaches 69.96% using 4096 time-steps. Both RMP-SNN and the baseline SNN with IF neurons are converted from our trained ANN with top-1 inference accuracy of 73.49%. The RMP-SNN with reduced threshold (green curve) reaches an accuracy of 68.93% using only 512 time-steps, which is 4.5 times faster than the baseline SNN with IF neurons using over 2300 time-steps. The RMP-SNN with reduced threshold (green curve) attains a spike rate as low as 1% throughout the entire inference time-steps.

The ResNet-20 ANN has been trained to have top-1 inference accuracy of 91.47% on CIFAR-10 dataset as shown in Fig.5 (in section 4.1). After conversion, the RMP-SNN reaches top-1 accuracy of 91.36% using 2048 time-steps, whereas the SNN with IF neurons reaches 90.45% using

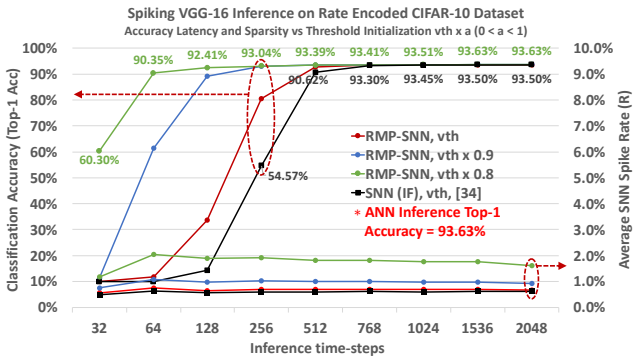

Figure 6. Inference performance comparison between the VGG-16 RMP-SNN and the baseline VGG-16 SNN (IF) on CIFAR-10 dataset.

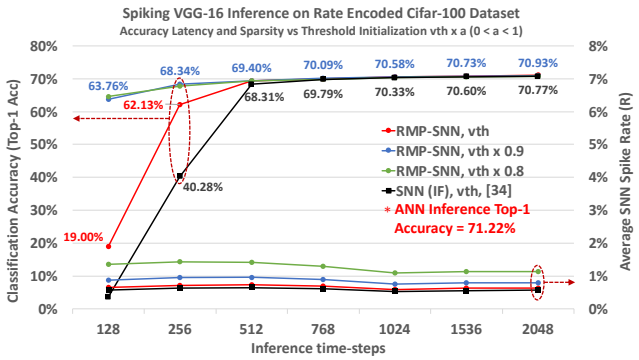

Figure 7. Inference performance comparison between the VGG-16 RMP-SNN and the baseline VGG-16 SNN (IF) on CIFAR-100 dataset.

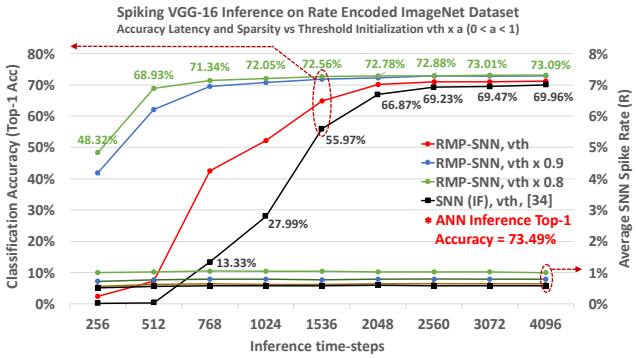

Figure 8. Inference performance comparison between the VGG-16 RMP-SNN and the baseline VGG-16 SNN (IF) on ImageNet dataset.

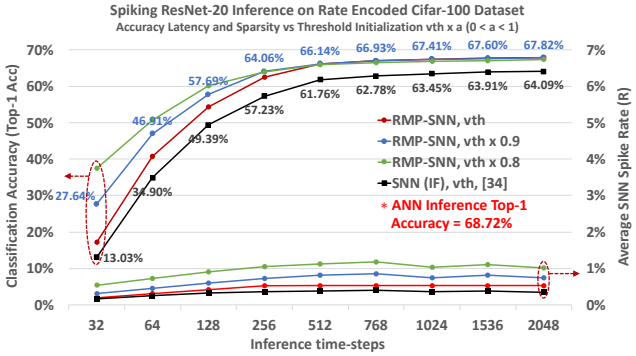

Figure 9. Inference performance comparison between the ResNet-20 RMP-SNN and the baseline ResNet-20 SNN (IF) on CIFAR-100 dataset.

the same 2048 time-steps. The RMP-SNN with reduced threshold (green curve) reaches an accuracy above 85% using only 64 time-steps, which is 8 times faster than the baseline SNN with IF neurons, that uses 512 time-steps. The RMP-SNN with reduced threshold (green curve) attains a spike rate around 2% throughout the inference time-steps.

The trained ResNet-20 ANN has top-1 inference accuracy of 68.72% on CIFAR-100 dataset as shown in Fig.9. The RMP-SNN reaches top-1 accuracy of 67.82% using 2048 time-steps, whereas the SNN with IF neurons reaches top-1 accuracy 64.09% using the same 2048 time-steps. The fastest RMP-SNN with reduced threshold (green curve) reaches an accuracy of 64.06% using only 256 time-steps, which is 8 times faster than the baseline SNN with IF neurons that uses about 2048 time-steps. The fastest RMP-SNN with reduced threshold (green curve) attains a spike rate about 1% throughout the inference time-steps.

The trained ResNet-34 ANN has top-1 inference accuracy of 70.64% on the ImageNet dataset as shown in Fig.10. The RMP-SNN reaches an accuracy of 69.89% using 4096 time-steps, whereas the SNN with IF neurons reaches 65.47% using the same 4096 time-steps. The fastest RMP-SNN with reduced threshold (green curve) reaches an

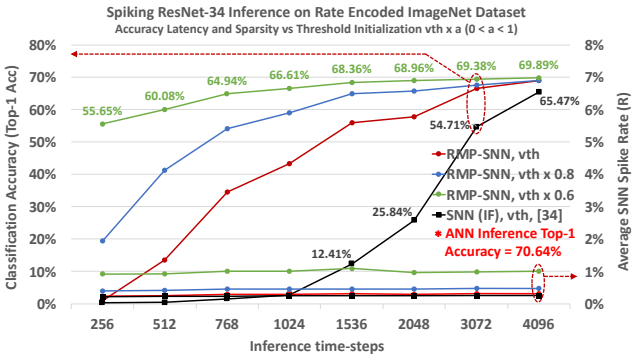

Figure 10. Inference performance comparison between the ResNet-34 RMP-SNN and the baseline ResNet-34 SNN (IF) on ImageNet dataset.

accuracy of 60.08% using only 512 time-steps, which is 7 times faster than the baseline SNN with IF neurons that uses more than 3500 time-steps. The fastest RMP-SNN with reduced threshold (green curve) attains a spike rate as low as 1% throughout the inference time-steps.

6. Conclusion and Discussion

In this work, we propose an ANN to SNN conversion technique. It uses novel spiking neuron model named RMP spiking neuron that retains a residual membrane potential after firing. The RMP spiking neuron better mimics the ReLU functionality than the IF neuron by allowing a residual potential to remain after the neuron has fired, alleviating the information loss that occurs during the ReLU to IF conversion. We also propose a threshold balancing technique which alleviates the spike rate vanishing issue in SNNs and significantly improved the latency and scalability of RMP-SNNs to very deep architectures. We implemented large scale deep network architectures such as VGG and Residual networks using the proposed conversion based training and evaluated performance on cifar-10, cifar-100 and ImageNet datasets. Our proposed RMP-SNNs achieve the best accuracies and lowest conversion loss than the state-of-the-art across all network architectures and datasets we tested.

Acknowledgment

This work was supported in part by C-BRIC, Center for Spintronic Materials, Interfaces, and Novel Architectures (C-SPIN), a MARCO and DARPA sponsored StarNet center, by the Semiconductor Research Corporation, National Science Foundation, Sandia National Laboratories, Vannevar Bush Faculty Fellowship and by the US Army Research Laboratory and the UK Ministry of Defense under Agreement Number W911NF-16-3-0001.

References

[1] Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert Legenstein, and Wolfgang Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. In *Advances in Neural Information Processing Systems*, pages 787–797, Montréal, Quebec, Canada, 2018. 1

[2] Peter Blouw, Xuan Choo, Eric Hunsberger, and Chris Eliasmith. Benchmarking keyword spotting efficiency on neuromorphic hardware. In *Proceedings of the 7th Annual Neuro-inspired Computational Elements Workshop*, page 1. ACM, 2019. 1

[3] Yongqiang Cao, Yang Chen, and Deepak Khosla. Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, 113(1):54–66, 2015. 2

[4] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018. 1

[5] Peter U Diehl and Matthew Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in computational neuroscience*, 9:99, 2015. 1

[6] Peter U Diehl, Daniel Neil, Jonathan Binas, Matthew Cook, Shih-Chii Liu, and Michael Pfeiffer. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2015. 2, 3, 5

[7] Peter U Diehl, Guido Zarrella, Andrew Cassidy, Bruno U Pedroni, and Emre Neftci. Conversion of artificial recurrent neural networks to spiking neural networks for low-power neuromorphic hardware. In *2016 IEEE International Conference on Rebooting Computing (ICRC)*, pages 1–8. IEEE, 2016. 2

[8] Steven K. Esser, Paul A. Merolla, John V. Arthur, Andrew S. Cassidy, Rathinakumar Appuswamy, Alexander Andreopoulos, David J. Berg, Jeffrey L. McKinstry, Timothy Melano, Davis R. Barch, Carmelo di Nolfo, Pallab Datta, Arnon Amir, Brian Taba, Myron D. Flickner, and Dharmendra S. Modha. Convolutional networks for fast, energy-efficient neuromorphic computing. *CoRR*, abs/1603.08270, 2016. 7

[9] Paul Ferré, Franck Mamalet, and Simon J Thorpe. Unsupervised feature learning with winner-takes-all based stdp. *Frontiers in computational neuroscience*, 12:24, 2018. 1

[10] Moritz Hardt and Tengyu Ma. Identity matters in deep learning. *CoRR*, abs/1611.04231, 2016. 6

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 1, 6

[12] David Heeger. Poisson model of spike generation. *Stanford University Handout*, 5:1–13, 2000. 3

[13] Eric Hunsberger and Chris Eliasmith. Spiking deep networks with lif neurons. *arXiv preprint arXiv:1510.08829*, 2015. 2

[14] Eric Hunsberger and Chris Eliasmith. Training spiking deep networks for neuromorphic hardware. *CoRR*, abs/1611.05141, 2016. 7

[15] Yingyezhe Jin, Wenrui Zhang, and Peng Li. Hybrid macro/micro level backpropagation for training deep spiking neural networks. In *Advances in Neural Information Processing Systems*, pages 7005–7015, Montréal, Quebec, Canada, 2018. 1

[16] Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, 2017. 1

[17] Saeed Reza Kheradpisheh, Mohammad Ganjtabesh, Simon J. Thorpe, and Timothée Masquelier. Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, 99:56–67, 2018. 1

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[19] Chankyu Lee, Syed Shakib Sarwar, Priyadarshini Panda, Gopalakrishnan Srinivasan, and Kaushik Roy. Enabling spike-based backpropagation in state-of-the-art deep neural network architectures. *arXiv preprint arXiv:1903.06379v3*, 2019. 1

[20] C. Lee, G. Srinivasan, P. Panda, and K. Roy. Deep spiking convolutional neural network trained with unsupervised spike timing dependent plasticity. *IEEE Transactions on Cognitive and Developmental Systems*, pages 1–1, 2018. 1

[21] Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in neuroscience*, 10:508, 2016. 1

[22] Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671, 1997. 1

[23] Timothée Masquelier and Simon J Thorpe. Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS computational biology*, 3(2):e31, 2007. 1

[24] Paul A Merolla, John V Arthur, Rodrigo Alvarez-Icaza, Andrew S Cassidy, Jun Sawada, Filipp Akopyan, Bryan L Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014. 1

[25] Milad Mozafari, Mohammad Ganjtabesh, Abbas Nowzari-Dalini, Simon J Thorpe, and Timothée Masquelier. Combining stdp and reward-modulated stdp in deep convolutional spiking neural networks for digit recognition. *arXiv preprint arXiv:1804.00227*, 2018. 1

[26] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010. 3

[27] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks. *arXiv preprint arXiv:1901.09948*, 2019. 1

[28] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal deep learn-

ing. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 689–696, 2011. 1

[29] Priyadarshini Panda and Kaushik Roy. Unsupervised regenerative learning of hierarchical features in spiking deep networks for object recognition. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 299–306, Vancouver, British Columbia, Canada, 2016. IEEE. 1

[30] José Antonio Pérez-Carrasco, Bo Zhao, Carmen Serrano, Begona Acha, Teresa Serrano-Gotarredona, Shouchun Chen, and Bernabé Linares-Barranco. Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate coding and coincidence processing–application to feedforward convnets. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2706–2719, 2013. 2

[31] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 1

[32] Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, and Michael Pfeiffer. Theory and tools for the conversion of analog to spiking convolutional neural networks. *arXiv preprint arXiv:1612.04052*, 2016. 2, 4

[33] Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in neuroscience*, 11:682, 2017. 2, 4, 6, 7

[34] Abhronil Sengupta, Yuting Ye, Robert Wang, Chiao Liu, and Kaushik Roy. Going deeper in spiking neural networks: Vgg and residual architectures. *Frontiers in neuroscience*, 13, 2019. 2, 3, 5, 6, 7

[35] Sumit Bam Shrestha and Garrick Orchard. Slayer: Spike layer error reassignment in time. In *Advances in Neural Information Processing Systems*, pages 1412–1421, Montréal, Quebec, Canada, 2018. 1

[36] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 6

[37] Gopalakrishnan Srinivasan, Priyadarshini Panda, and Kaushik Roy. Stdp-based unsupervised feature learning using convolution-over-time in spiking neural networks for energy-efficient neuromorphic computing. *J. Emerg. Technol. Comput. Syst.*, 14(4):44:1–44:12, Nov. 2018. 1

[38] Gopalakrishnan Srinivasan and Kaushik Roy. Restocnet: Residual stochastic binary convolutional spiking neural network for memory-efficient neuromorphic computing. *Frontiers in Neuroscience*, 13:189, 2019. 1

[39] Amirhossein Tavanaei, Zachary Kirby, and Anthony S Maida. Training spiking convnets by stdp and gradient descent. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Rio de Janeiro, Brazil, July 2018. 1

[40] Johannes C. Thiele, Olivier Bichler, and Antoine Dupret. Event-based, timescale invariant unsupervised online deep learning with stdp. *Frontiers in Computational Neuroscience*, 12:46, 2018. 1

[41] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in neuroscience*, 12:331, 2018. 1

[42] Bo Zhao, Ruoxi Ding, Shoushun Chen, Bernabe Linares-Barranco, and Huajin Tang. Feedforward categorization on aer motion events using cortex-like features in a spiking neural network. *IEEE transactions on neural networks and learning systems*, 26(9):1963–1978, 2014. 2