



OPEN ACCESS

EDITED BY

Teresa Serrano-Gotarredona,
Spanish National Research Council
(CSIC), Spain

REVIEWED BY

Chen Li,
King's College London, United Kingdom
Sashmita Panda,
Indian Institute of Technology Kharagpur, India

*CORRESPONDENCE

Andrea Castagnetti
✉ andrea.castagnetti@univ-cotedazur.fr

SPECIALTY SECTION

This article was submitted to
Neuromorphic Engineering,
a section of the journal
Frontiers in Neuroscience

RECEIVED 30 January 2023

ACCEPTED 14 February 2023

PUBLISHED 03 March 2023

CITATION

Castagnetti A, Pegatoquet A and Miramond B
(2023) Trainable quantization for Speedy Spiking
Neural Networks. *Front. Neurosci.* 17:1154241.
doi: 10.3389/fnins.2023.1154241

COPYRIGHT

© 2023 Castagnetti, Pegatoquet and
Miramond. This is an open-access article
distributed under the terms of the [Creative Commons Attribution License \(CC BY\)](https://creativecommons.org/licenses/by/4.0/). The use,
distribution or reproduction in other forums is
permitted, provided the original author(s) and
the copyright owner(s) are credited and that
the original publication in this journal is cited, in
accordance with accepted academic practice.
No use, distribution or reproduction is
permitted which does not comply with these
terms.

Trainable quantization for Speedy Spiking Neural Networks

Andrea Castagnetti*, Alain Pegatoquet and Benoît Miramond

LEAT, Université Côte d'Azur, CNRS, Sophia Antipolis, France

Spiking neural networks are considered as the third generation of Artificial Neural Networks. SNNs perform computation using neurons and synapses that communicate using binary and asynchronous signals known as spikes. They have attracted significant research interest over the last years since their computing paradigm allows theoretically sparse and low-power operations. This hypothetical gain, used from the beginning of the neuromorphic research, was however limited by three main factors: the absence of an efficient learning rule competing with the one of classical deep learning, the lack of mature learning framework, and an important data processing latency finally generating energy overhead. While the first two limitations have recently been addressed in the literature, the major problem of latency is not solved yet. Indeed, information is not exchanged instantaneously between spiking neurons but gradually builds up over time as spikes are generated and propagated through the network. This paper focuses on quantization error, one of the main consequence of the SNN discrete representation of information. We argue that the quantization error is the main source of accuracy drop between ANN and SNN. In this article we propose an in-depth characterization of SNN quantization noise. We then propose a end-to-end direct learning approach based on a new trainable spiking neural model. This model allows adapting the threshold of neurons during training and implements efficient quantization strategies. This novel approach better explains the global behavior of SNNs and minimizes the quantization noise during training. The resulting SNN can be trained over a limited amount of timesteps, reducing latency, while beating state of the art accuracy and preserving high sparsity on the main datasets considered in the neuromorphic community.

KEYWORDS

Spiking Neural Networks, quantization error, low latency, sparsity, direct training

1. Introduction

The field of neuromorphic engineering, especially Spiking neural networks (SNNs), is emerging as a new paradigm for the design of low-power and real-time information processing hardware (Abderrahmane et al., 2020). The spike information coding used by SNNs enables sparse and event-based computation through the network. The combination of these properties may lead to more energy efficient hardware implementations of neural networks, allowing state-of-the-art AI algorithms to be executed on mobile platforms with a reduced power budget (Mendez et al., 2022). However, to achieve these energy gains while simultaneously reaching the level of performance of Artificial Neural Networks (ANNs), SNNs must be able to encode analog data with high precision using very compact codes, i.e., spike trains. The encoding precision in SNN is directly related to the latency of the network. Increasing the conversion time, thus generating more spikes, lowers the quantization errors

and improves performance at the cost of energy overhead. The trade-off between conversion time, i.e., latency, and performance, is an increasingly active area of research (Li et al., 2021, 2022) and the main subject of this paper. Training methods for low latency and high precision SNN can be divided in two categories: ANN-SNN conversion and direct training. ANN-SNN conversion generally lead to SNN with no accuracy degradation. However, this comes at the cost of increasing latency. At the opposite, direct training methods feature low latency, but suffered from accuracy degradation, especially on deep neural networks such as ResNet (Fang et al., 2021; Li et al., 2022).

In this paper, we propose a direct training method that can achieve both low latency and high precision thanks to Adaptive Threshold Integrate and Fire (ATIF) neurons. ATIF direct training minimizes accuracy loss compared to ANN by applying a novel trainable quantization scheme. By efficiently compressing information we can achieve high accuracy with few timesteps even for deep networks.

The main contributions of our work are listed below:

- **Quantization noise in SNN:** Spiking neurons quantize information by converting their analog inputs into sequences of spikes. We characterize the quantization error and its relationship to the different parameters of a spiking neuron.
- **Information compression through trainable quantization:** We propose a learning approach that reduces quantization error by adapting the neuron's parameters during training and using a new neural model called ATIF.
- **Low latency and sparse SNN:** We validate our approach on different image and audio classification problems, thus defining new state of the art results in terms of accuracy and latency. Moreover we show that these performance can be achieved with a significant level of sparsity. Specifically, we achieve 94.65% accuracy on CIFAR-10, and 94.31% on Google Speech Commands with less than one spike per neuron.

2. State of the art

In the last few years, the development of SNN has been driven by the need of matching the performance of the ANN on complex image processing tasks. Early works focused on unsupervised or semi-supervised learning algorithms based on spike timing dependent plasticity (STDP) (Diehl and Cook, 2015; Srinivasan et al., 2018). However, networks trained with STDP yield in general to considerable lower accuracy than ANN or SNN trained with backpropagation.

To take advantage of better performance provided by supervised learning, several methods have been developed to convert ANNs, trained using standard schemes like backpropagation, into SNNs for event-driven inference (Diehl et al., 2015; Rueckauer et al., 2017). The ANN-SNN conversion is based on the idea that firing rates of spiking neurons should match the activations of analog neurons. Early demonstration on complex dataset like Imagenet or CIFAR-10 showed that SNNs almost match the accuracy of ANN but at the cost of a higher

latency. As an example, Sengupta et al. (2019) was able to achieve competitive results on CIFAR-10 with a latency of 2,500 timesteps.

Han et al. (2020) proposed a conversion-based training using *soft-reset* spiking neurons. The Integrate and Fire (IF) with soft-reset neuron implements a uniform quantization scheme between its analog input and its spiking output, thus leading to a reduced quantization error compared to hard-reset neurons. Moreover, to ensure that spiking neurons operate in the linear regime the authors proposed a technique to balance the firing threshold (V_{th}). With the proposed model, the authors were able to achieve an accuracy of 60.30% with a latency of 32 timesteps on CIFAR-10 using a VGG-16 network. In our work, we also use the IF with soft-reset neuron model to take advantage of its uniform quantization.

Ding et al. (2021) applied a clipped ReLU during ANN training to better emulate the behavior of spiking neurons. The clipping point, which is the equivalent to the firing threshold of a spiking neuron, is trained layer-wise. After conversion, the authors obtain an accuracy of 85.40% with a latency of 32 timesteps on CIFAR-10 using a VGG-16 network.

Previous works have shown that threshold balancing clearly helps reducing the quantization error, thus decreasing the accuracy loss of SNNs. However, they fail to convert an ANN into an SNN within extremely low time steps, where quantization errors are higher. To overcome this issue, Li et al. (2021) proposed a post-training calibration pipeline that fine-tunes, layer-by-layer, the network parameters, including weights, bias and membrane potentials, therefore minimizing the local conversion error (i.e., quantization error). An accuracy of 86.57% at 4 timesteps was then obtained on CIFAR-10 using a VGG-16 network.

The authors of Li et al. (2022) went one step beyond by proposing to convert a quantized ANN to an SNN. They use a quantization-aware-training method called Learned step size quantization (LSQ) (Esser et al., 2020) to train a quantized ANN. In order to transfer the weights from the quantized ANN to the SNN, the spiking neuron model was modified to match the response curve of the quantized ReLU. However, for this method to be effective, the proposed spiking neuron must be able to generate spikes with negative polarity, which is not biologically plausible. With the previous conversion method, an accuracy of 92.64% at 4 timesteps was obtained on CIFAR-10 using a VGG-16 network. Beyond the excellent accuracy score, the authors of Li et al. (2022) have shown that there is an equivalence between quantized ANN and SNNs. Moreover, to obtain state of the art results, quantization aware training must be used to jointly optimize accuracy and quantization error.

Another method to obtain low latency SNNs is to train the spiking network directly by surrogate gradients (Neftci et al., 2019). Here, a surrogate function is used, during gradient back-propagation, to replace the binary non-linearity of spiking neurons. This allows gradient flowing thus making back-propagation possible in the spiking domain. Direct training can optimize at the same time, the network accuracy and the quantization error introduced by the spiking neurons. It can therefore be considered as a spiking-specific form of quantization aware training.

Rathi and Roy (2021) use direct learning to fine tune network parameters transferred from an ANN. They are able to achieve an accuracy of 92.70% at 5 timesteps on CIFAR-10 with VGG-16.

The authors of Fang et al. (2021) used direct learning to train ResNet without any previous conversion from ANN. They proposed SEW-ResNet, a spiking adaptation of ResNet that overcomes the vanishing/exploding gradient problem that occurs with directly trained spiking deep residual networks. They achieved an accuracy of 67.04% at 4 timesteps on Imagenet with a ResNet34. Their results show that it is possible to train deep spiking neural networks and obtain very competitive results on complex datasets.

Finally, in our work we make use of computationally efficient spiking neuron models that approximate the behavior observed in real neurons. In these models the action potentials, i.e., the spikes, are approximated using binary pulses of infinitesimally short duration. However, recent studies have shown that biological neurons can generate several spiking dynamics and they can fire spikes of different amplitudes and duration (Chakraborty et al., 2022). Moreover, in contrast to the structure of the human cortex (Panda et al., 2021), we only consider networks composed of layers of neurons with identical characteristics. The integrate-and-fire model used in this paper does take into account only a limited set of features of the biological neuron. However, it is compact and computationally efficient, thus well-suited for SNNs in the context of the machine learning tasks that we target in our work.

In this paper we propose to use direct learning to jointly optimize the network and the spiking neurons parameters. Moreover, we show that it is possible to reach or outperform state of the art results without using any non-biologically plausible artifacts, like negative polarity or non-binary spiking signals. All the results presented in this Section will be summarized in Tables 1, 2 in Section 4.

3. Methods

3.1. Spike based information compression

Considering an n -layer fully connected or convolutional ANN, the output of the layer l can be described as:

$$y^l = h(x^l W^l + b^l), \quad l \in [1, n] \quad (1)$$

Where, x^l , W^l , b^l are the input activation, the weights and the bias of the layer l , respectively. Moreover, $h(\cdot)$ denotes the ReLU activation function. In SNNs the activation function is replaced with a spiking neuron, whose role is to implement the ReLU non-linearity ($\max(0, x)$) and discretize its input signal into spikes. Here, we use the Integrate-and-Fire (IF) neuron model with soft-reset that can be described, at each timestep t , by the following equations:

$$H^l(t) = V^l(t-1) + i^l(t) \quad (2)$$

$$i^l(t) = z^{l-1}(t) W^l + b^l \quad (3)$$

$$z^l(t) = \Theta(H^l(t) - V_{th}) \quad (4)$$

$$V^l(t) = H^l(t)(1 - z^l(t)) + (H^l(t) - V_{th})z^l(t) \quad (5)$$

Where, $H^l(t)$ and $V^l(t)$ represents the membrane potential after the input integration and after the reset operation that follows the spike emission at time t , respectively. The spiking output at time t is represented by $z^l(t)$. Here, we stress the fact that $z^l(t)$ can only have binary values. The input of the spiking neuron, $i^l(t)$, can be

expressed as the product of the weights with the binary signal, i.e., the spikes, generated by the preceding layer plus a constant bias. As can be seen from Equations (2) and (3), the product can be replaced with an addition of the weights with the membrane potential whenever $z^{l-1}(t) = 1$. Moreover, the bias is added to the membrane potential at each timestep regardless the value of $z^{l-1}(t)$. Equations (4) and (5) describe the generation of a spike and the soft-reset operation, respectively. The function $\Theta(\cdot)$ represents the Heaviside step function.

The SNN forward operation described by the previous equations, is repeated through T timesteps. The output of the spiking layer l can be decoded as follows:

$$y_s^l = \frac{1}{T} \sum_{t=1}^T z^l(t) \quad (6)$$

Equation (6) defines the decoding scheme for a rate-coded network. In this scheme, the information is coded by the number of spikes generated by a spiking neuron over a fixed length of time T , i.e., the firing-rate. Since a spiking neuron generates spikes proportionally to its input current, the time average described in Equation (6) will converge to y^l , when $T \rightarrow \infty$. Moreover, the time integration of $z^l(t)$ which is a binary variable, leads to a quantization of the decoded output y_s^l . Using the preceding equations, it is possible to show that the quantization function of a IF with soft-reset spiking neuron can be expressed in the following closed form:

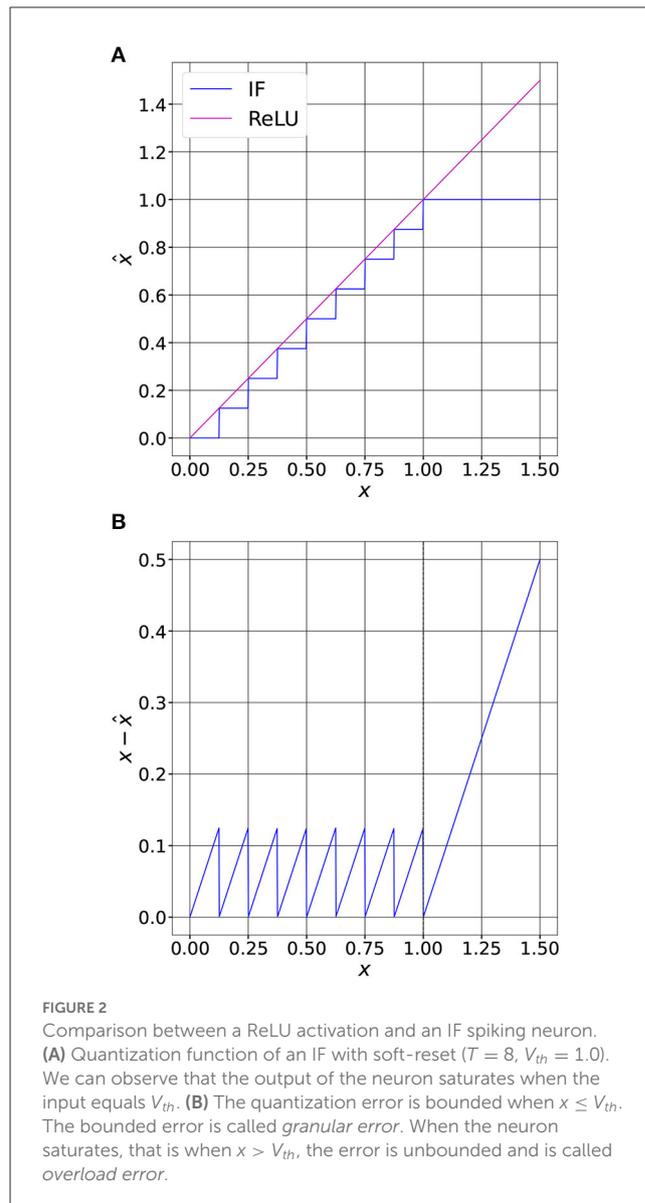
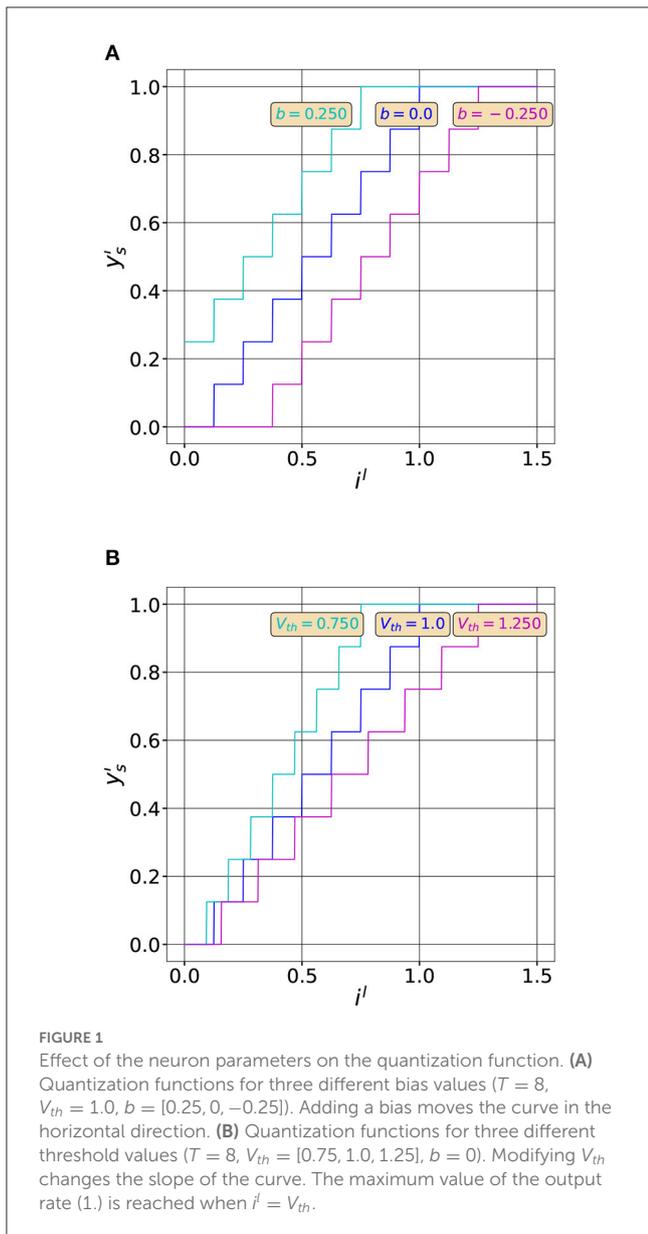
$$\sum_{t=1}^T z(t) = \min \left\{ T, \frac{\sum_{t=1}^T (z^{l-1}(t) W^l) + b T}{V_{th}} \right\} \quad (7)$$

The numerator of Equation (7), $\sum_{t=1}^T (z^{l-1}(t) W^l) + b T$, represents the integral of the input current, noted in the following i^l , over T timesteps. Figure 1 shows the effect of the different neuron parameters on the quantization function.

As we can observe, by varying the neuron parameters we can modify the quantization curve. For example, the rational behind threshold balancing is to adapt the quantization range, by modifying V_{th} to match the input distribution as shown in Figure 1B. We can also observe that the quantization function has exactly $T + 1$ uniform quantization intervals. We therefore expect a lower quantization noise by increasing the latency of the network, that is T . However, increasing the latency hinders the computational efficiency of SNNs. So, in order to efficiently compress information with spiking neurons and maintain, at the same time, their computational efficiency we have to reduce the quantization noise without increasing T . This can be achieved by optimizing the quantization function of spiking neurons to better match the input distribution of i^l as we will see in the next sections.

3.2. Quantization error analysis

Here, we characterize the quantization noise introduced by the spiking neurons. The effect of the neuron parameters and the input distribution on the quantization process are first studied. Let us consider an analog signal x and its quantized version \hat{x} . We define the quantization noise introduced during the conversion process



using the Signal-to-Quantization-Noise-Ratio (SQNR) defined below:

$$SQNR(x) = 10 \log_{10} \left(\frac{\mathbb{E}[x^2]}{\mathbb{E}[(x - \hat{x})^2]} \right) \quad (8)$$

In ANNs the ReLU activation function does not introduce any quantization noise. So, if we consider that x is the input of a ReLU, then $y = ReLU(x) = x$ when $x \geq 0$. At the opposite in SNNs, the input current of a spiking neuron is quantized as shown in Equation (7). If we define $x = i^l$ as the input of the neuron, the decoded spiking output $y_s = IF(x) = \hat{x}$ is the quantized representation of its input. This process is shown in Figure 2.

From Figure 2B, we can observe that when the input of the neuron is lower than V_{th} , the quantization error is bounded. At the opposite, when $x > V_{th}$, the quantization error can grow without any bounds. To minimize the quantization error within a uniform

quantization scheme it is thus necessary to set the saturation point, that is V_{th} , to balance these two sources of error. To do so, we must know the probability density function (PDF) of the input, then optimize V_{th} to reduce the quantization noise, as we will see in the next section.

3.3. PDF-optimized quantization for spiking neurons

Let us consider an IF spiking neuron described by the Equations (2)–(4). In this paper, the surrogate gradient method is used to compute the derivative of the Heaviside step function during error back-propagation, that is $\Theta'(x) = \sigma'(x)$. Where, $\sigma(x)$ denotes the surrogate function, i.e., an approximation of the step function. Throughout our paper, we use the sigmoid ($\sigma(x) = \frac{1}{1+e^{-x}}$) as surrogate function.

Let V_{th} be a trainable parameter of the spiking neuron, i and z its input and output, respectively. Let us call $\frac{\partial L}{\partial z}$ the upstream gradient to the neuron. The downstream gradient with respect to V_{th} can then be computed using the chain rule by multiplying the upstream gradient by the local gradient with respect to V_{th} . For the sake of notation simplicity we define $q = (H(t) - V_{th})$, the input of the Heaviside step function. We can then compute the downstream gradient as follows:

$$\frac{\partial L}{\partial V_{th}} = \frac{\partial L}{\partial z} \cdot \frac{\partial z}{\partial q} \cdot \frac{\partial q}{\partial V_{th}} \tag{9}$$

Using Equation (4) and the derivative of the surrogate function we can compute the gradient of z with respect to q as follows:

$$\frac{\partial z}{\partial q} = \sigma'(q) \tag{10}$$

Then, from the definition of q we can determine:

$$\frac{\partial q}{\partial V_{th}} = -1 \tag{11}$$

Finally, by replacing Equations (10) and (11) into Equation (9) we obtain the approximation of the downstream gradient:

$$\frac{\partial L}{\partial V_{th}} = -\frac{\partial L}{\partial z} \cdot \sigma'(q) \tag{12}$$

The neuron output, denoted \hat{y} , is computed by decoding the spiking sequence after T timesteps. We would like to find the value of V_{th} that minimizes the quantization error of the neuron. To do so, we analyze the optimization process for a single spiking neuron whose input current i follows a Gaussian distribution with zero mean, $i \sim \mathcal{N}(0, \sigma^2)$. We first compare trainable and fixed V_{th} neurons with a fixed number of timesteps $T = 4$. The loss function is defined as the RMS error between the input and the output of the neuron:

$$L(i, \hat{y}) = \mathbb{E}[(i - \hat{y})^2] \tag{13}$$

As we can observe from Equation (13), by minimizing the loss function we maximize the SQNR for a given input distribution. We simulated the above optimization problem for both an IF neuron with a fixed $V_{th} = 1$ and an IF neuron with its threshold modified during the learning process. At each iteration, an input i is drawn from a Gaussian distribution with $\sigma = 0.1$. The neuron then converts the input into spikes, that are finally decoded to obtain the estimate \hat{y} . Once the loss is computed with Equations (13), the error is back-propagated and V_{th} (the only neuron parameter), is updated using standard gradient-based optimization (Adam optimizer, $l_r = 10^{-3}$). In the following this model is denoted as ATIF-u.

The threshold voltage is initialized to the value 1 for all neurons. The resulting SQNR is shown in Figure 3A. As it can be observed, the SQNR at the beginning of the optimization process is the same for both neurons models. While the quantization error does not vary for the model with a fixed V_{th} , optimizing V_{th} can increase by more than 10 dB the SQNR compared to a fixed V_{th} model, for the same amount of timesteps ($T = 4$). From Figure 3B, we can observe that the gain in SQNR is obtained by decreasing V_{th} ,

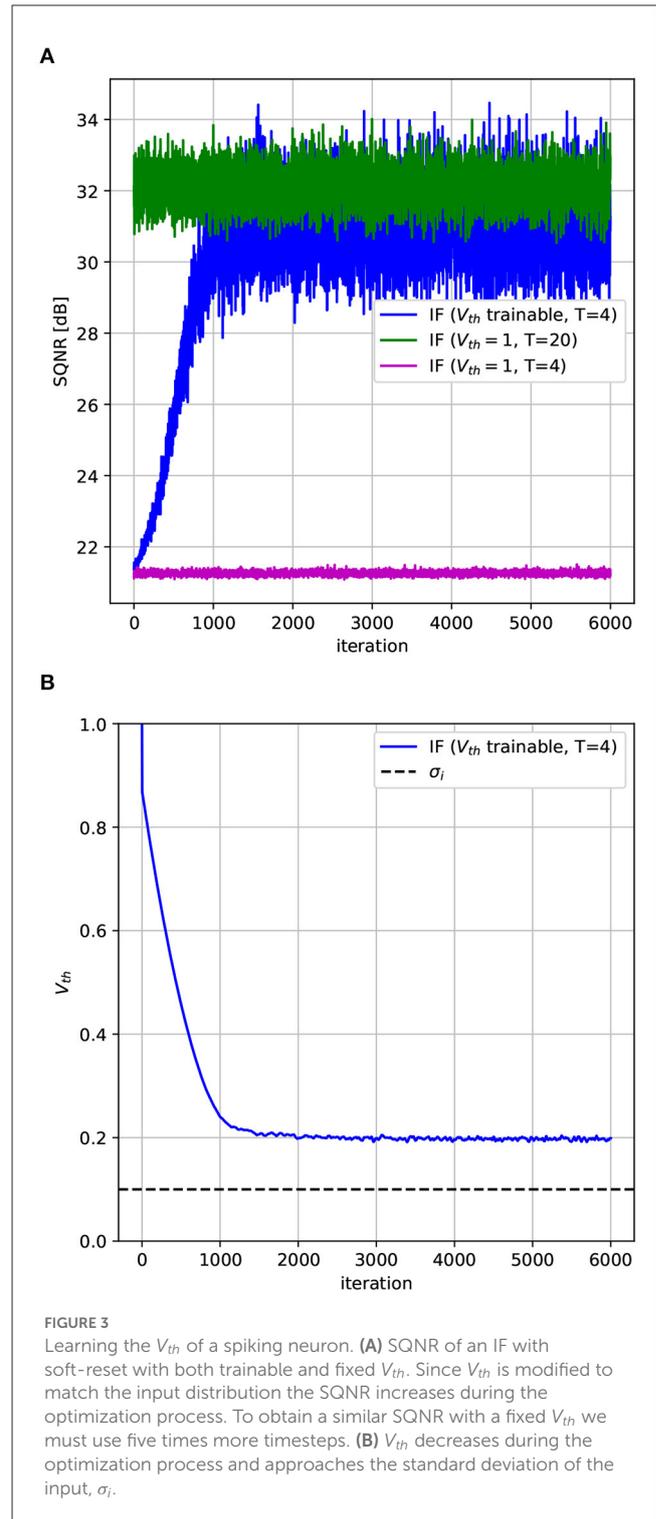
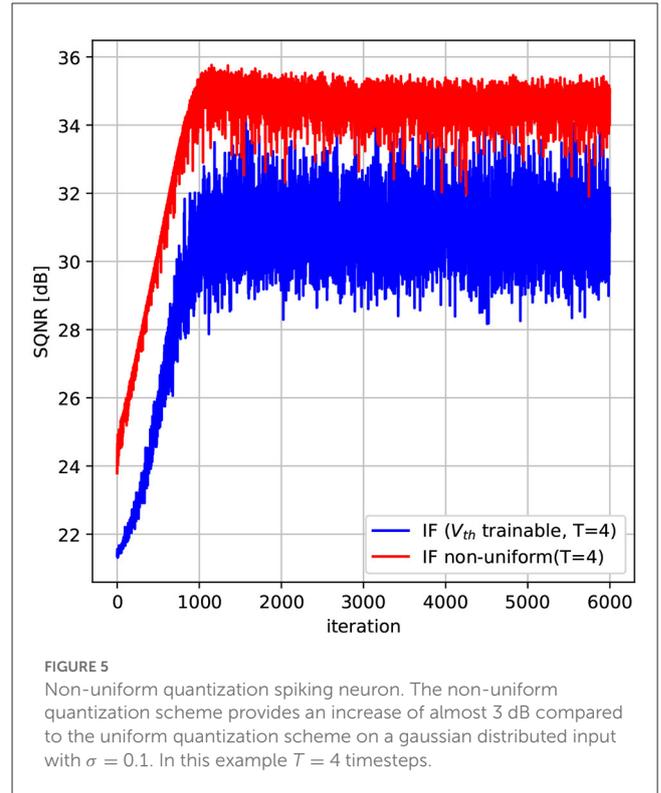
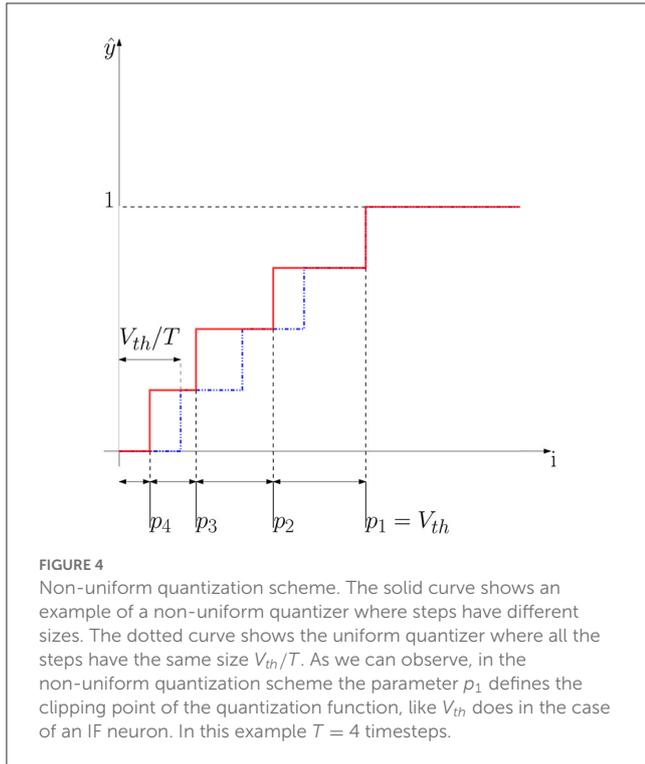


FIGURE 3 Learning the V_{th} of a spiking neuron. **(A)** SQNR of an IF with soft-reset with both trainable and fixed V_{th} . Since V_{th} is modified to match the input distribution the SQNR increases during the optimization process. To obtain a similar SQNR with a fixed V_{th} we must use five times more timesteps. **(B)** V_{th} decreases during the optimization process and approaches the standard deviation of the input, σ_i .

thus moving the quantization intervals near the region where most of the input values fall, in our case $[0, 2\sigma_i]$. By optimizing the quantization function of the spiking neuron we have then been able to significantly decrease the quantization error without increasing the timesteps. As an example, to obtain the same SQNR with a fixed V_{th} , we should have used five times more timesteps, that is $T = 20$, as it can be observed from Figure 3A.



We have obtained the previous results by assuming that the input of the neurons is normally distributed, which is a reasonable hypothesis for SNN and ANN in general. However, similar results and conclusions can be obtained for different type of distributions. The optimization procedure described above, that can be integrated into a Quantization-aware-training framework for SNN, is similar to some extent to what LSQ (Esser et al., 2020) did for quantized ANNs. That is, a uniform quantization scheme is tuned to match the input distribution of the neurons. We can indeed further reduce the quantization error, beyond what is possible with a uniform quantization scheme, by modifying the size of the quantization steps, thus making the quantizer non-uniform. We propose, in the next section, a modified spiking neuron model able to implement this quantization strategy.

3.4. PDF-optimized non-uniform quantization for spiking neurons

In the uniform quantization scheme described in the previous section, the size of each quantization step is equal to V_{th}/T , as can be observed from Figures 1, 2. Since V_{th} is constant and does not vary during time, the number of spikes generated by the neuron only depends on the input amplitude.

Figure 4 shows an example of a non-uniform quantization scheme. We can obtain this scheme by modifying the spiking neuron, precisely, we let V_{th} change during time. Let us start with an example where $T = 4$. In this case, we would like to obtain exactly four values of $V_{th}(t) = [V_{p_1}, V_{p_2}, V_{p_3}, V_{p_4}]$. In the first interval, that is when $i \geq p_4$, the neuron outputs only one spike at the last timestep. Let us denote this particular output with the following

notation: $z(t) = [0001]$. We can compute the threshold of the last timestep (V_{p_4}) as follows:

$$V_{p_4} = p_4 T \tag{14}$$

By setting $V_{th}(t = 4) = V_{p_4}$ we constrain the neuron to generate a spike at the last timestep whenever an input of amplitude p_4 is presented at the neuron input during T timesteps. We have thus set the quantization step at the value $i = p_4$. Let us now consider the quantization step at value p_1 . When the input $i \geq p_1$, the neuron has to output the maximum rate, that is $z(t) = [1111]$. This condition allows us to define the threshold for the first timestep as follows:

$$V_{p_1} = p_1 \tag{15}$$

Following the same reasoning, when $p_3 \leq i \leq p_2$ the neuron generates a spikes in the last two timesteps, that is $z(t) = [0011]$. This condition can be written as follows:

$$\begin{cases} (T - 1)p_3 \geq V_{p_3}, & t = T - 1 \\ (T - 1)p_3 - V_{p_3} + p_3 = V_{p_4}, & t = T \end{cases} \tag{16}$$

The first equation in 16 describes the state of the membrane potential of the neuron at timestep $T - 1$. We set the neuron threshold at the penultimate timestep to be equal to V_{p_3} to make the neuron fire. Following a spike emission at timestep $T - 1$, the membrane potential is soft-reset, then the input p_3 is accumulated at timestep T as shown in the second equation in 16. The equation system shown in 16 has the following solution:

$$V_{p_3} = T(p_3 - p_4) \tag{17}$$

In the same way, when $p_2 \leq i \leq p_1$, the neuron will output the following sequence $z(t) = [0111]$ then the following three conditions are met:

$$\begin{cases} (T-2)p_2 \geq V_{p_2}, & t = T-2 \\ (T-2)p_2 - V_{p_2} + p_2 \geq V_{p_3}, & t = T-1 \\ (T-2)p_2 - V_{p_2} + p_2 - V_{p_3} + p_2 = V_{p_4}, & t = T \end{cases} \quad (18)$$

Which leads us to the following solution:

$$V_{p_2} = (T-1)p_2 - T(p_3 - p_4) \quad (19)$$

In the following this model is denoted as ATIF-nu. We have simulated the proposed neuron model for a gaussian distributed input with $\sigma = 0.1$. The model has T trainable parameters, namely $[p_1, \dots, p_T]$. The thresholds are then computed using Equation (15) to Equation (19), for the case $T = 4$. We use the same optimization procedure and loss described in Section 3.3. The simulation results are shown in Figure 5 along with the uniform quantization neuron with trainable V_{th} .

As it can be observed, using a non-uniform quantization scheme can increase the SQNR by almost 3dB compared to the uniform case, without increasing the number of timesteps. This gain comes from the fact that using non-uniform quantization steps, we can better approximate the input data in regions that have more probability mass. As the input of neurons in SNNs can

often be modeled as gaussian sources peaked near zero, we expect that a non-uniform quantization scheme can help improving the performance of SNNs as we will see in the next section.

4. Experiments and results

4.1. Experimental setup

We trained both VGG-16 and ResNet-18 models using direct training on two different image classification problems with increasing complexity: CIFAR-10, CIFAR-100. We trained both networks with the neuron models described in Sections 3.3 and 3.4. In our setup, the threshold of neurons are trained layer-wise, so that all the spiking neurons of a given layer share the same V_{th} . For each model and dataset, we also trained a formal version of the network, where spiking neuron are replaced with ReLU activation to compare SNN with a full precision ANN. Both SNN and ANN were trained using stochastic gradient descent (SGD), with a learning rate of $8 \cdot 10^{-2}$. The learning rate is exponentially decayed with a factor of 0.9 each 30 epochs. Each network is trained for 900 epochs. In SNNs, the input is analog coded (Rueckauer et al., 2017), that is the spiking neurons of the first layer receive a constant input current. We use data augmentation (random resize and horizontal flip) as well as mixup with $\alpha = 1$.

TABLE 1 Benchmark results on CIFAR-10/100 datasets.

Method	Architecture	ACC (ANN)	ACC (SNN)	Latency	θ
CIFAR-10					
RMP (Han et al., 2020)*	VGG-16	93.63	60.3	32	-
	ResNet-20	91.47	91.36	2048	-
ACP (Li et al., 2021)*	VGG-16	95.6	86.57	4	-
	ResNet-20	96.72	84.70	4	-
QFFS (Li et al., 2022)*	VGG-16	92.44 (2/3 bits)	92.64	4	-
	ResNet-18	93.12 (2/3 bits)	93.14	4	-
ATIF-u†	VGG-16	95.6	92.51	4	0.111
	ResNet-18	95.96	93.84	4	0.113
ATIF-nu†	VGG-16	95.6	93.13	4	0.129
	ResNet-18	95.96	94.65	4	0.148
CIFAR-100					
RMP* (Han et al., 2020)*	VGG-16	71.22	63.76	128	-
	ResNet-20	68.72	67.82	2048	-
ACP* (Li et al., 2021)*	VGG-16	77.93	55.60	4	-
	ResNet-20	81.51	54.96	4	-
ATIF-u†	VGG-16	74.47	66.54	4	0.159
	ResNet-18	74.35	71.42	4	0.192
ATIF-nu†	VGG-16	74.47	66.92	4	0.167
	ResNet-18	74.35	70.83	4	0.191

Benchmark results on CIFAR-10/100 datasets. Best accuracy results for both ResNet and VGG-16 networks are highlighted in bold. Symbol * denotes ANN-to-SNN conversion methods while † denotes surrogate gradient learning, i.e., direct training methods.

TABLE 2 Benchmark results on Google Speech Commands V2 dataset (35 classes).

Method	Architecture	ACC (ANN)	ACC (SNN)	Latency	θ
Recurrent (Cramer et al., 2022) [†]	RSNN	-	50.9	200	-
E2E (Yang et al., 2022)*	ResNet-8	-	92.9	32	-
ATIF-u [†]	ResNet-18	94.46	94.31	4	0.12
ATIF-nu [†]	ResNet-18	94.46	94.29	4	0.12

Best accuracy results are highlighted in bold. Symbol * denotes ANN-to-SNN conversion methods while [†] denotes surrogate gradient learning, i.e., direct training methods.

TABLE 3 The impact of V_{th} on the accuracy and the sparsity of the SNN.

Quantizer	V_{th}	ACC (SNN)	Latency	θ	Spikes/neuron
Uniform	Fixed	89.68	4	0.17	0.68
ATIF-u	Trainable	93.84	4	0.113	0.452
ATIF-nu	Trainable	94.65	4	0.148	0.592

Results are given for a ResNet-18 network on the CIFAR-10 dataset.

In addition to the image classifications problems described above, we also carried out experiments on a keyword spotting (KWS) dataset. In an automatic speech recognition system, KWS consists in detecting a relatively small set of predefined keywords. We used Google Speech Commands (GSC) (Warden, 2018) V2, a dataset of audio signals sampled at 16 kHz composed of 1-s recordings of 35 spoken keywords. Raw audio signals are pre-processed to extract Mel Frequency Cepstral Coefficients (MFCC). We used 10 MFC Coefficients, FFT of size 1024, a window size of 640 with a hop of 320, and a padding of 320 on both sides. The pre-processing generates a 48×10 coefficients matrix that is subsequently processed by the neural networks. We trained a Resnet-18 network using the same training configuration used for CIFAR-10/100 except for the learning rate and the number of epochs. The learning rate is initialized at 10^{-3} and exponentially decayed with a factor of 0.1 each 20 epochs. Finally, each network is trained for 80 epochs. We used PyTorch and the SpikingJelly (Fang et al., 2020) framework for simulating SNNs. In the following sections, we report the accuracy as well as the latency and the sparsity of the SNN. We measured the sparsity of our networks, that we call θ , by counting the average number of spikes generated by the spiking neurons during the inference. The sparsity of a tensor of size (m, n) is computed as follows:

$$\theta = \frac{\sum_{t=1}^T \sum_{i=0}^n \sum_{j=0}^m z_{ij}(t)}{n \times m \times T} \quad (20)$$

To report a single global sparsity, we average the sparsity of each tensor in the SNNs. Moreover, the sparsity is averaged over the entire test set that is composed of 10K images in the case of CIFAR-10. The θ parameters measure the average activity of spiking neurons. A low value means low activity and therefore a potential increase in energy efficiency of the network when deployed on a neuromorphic dedicated circuit (Lemaire et al., 2022).

4.2. Experimental results on CIFAR-10/100

Benchmark results on CIFAR-10/100 datasets are shown in Table 1. Our models perform consistently better than recent state of the art ANN-SNN conversion methods on both datasets. As an example, we improve the top-1 accuracy of ResNet-18 on CIFAR-10 by 1.5% with respect to QFFS (Li et al., 2022). Experimental results also show that our quantization approaches outperform previous methods regardless of the network architecture. As an example, we improve the top-1 accuracy on CIFAR-10 for both ResNet-18 and VGG-16 architectures. The non-uniform quantization scheme provides the best accuracy scores in 3 out of four configurations, while the uniform scheme provides a slightly better accuracy score on ResNet-18 and CIFAR-100. It is worth noting that, these improvements are obtained using only 4 timesteps. So, we are able to improve the classification accuracy without degrading the SNN latency compared to current state of the art methods. Besides the latency, the sparsity parameter also has a strong impact on the SNN efficiency. The amount of operations executed during SNN inference is indeed related to the average firing rate of the neurons (Lemaire et al., 2022). The sparsity parameter, θ , shows that our networks are able to classify images using on average less than one spike per neuron. As an example, for VGG-16 with a uniform quantization scheme, each neuron generates on average $0.111 \times 4 = 0.44$ spikes. It can also be observed that the non-uniform quantization scheme generates more spikes on average than the uniform quantizer. Since more quantization steps are allocated on the region of the input where data appears more frequently, i.e., where the PDF has higher probability mass, the generation of spikes increases accordingly.

4.3. Experimental results on Google Speech Commands

The experimental results on the Google Speech Commands dataset are shown in Table 2. We compared both our quantization

schemes, ATIF-u and ATIF-nu, with two recently published SNNs. The first network, called Recurrent (Cramer et al., 2022), is a single layer recurrent-SNN composed of 128 Leaky-Integrate-and-Fire (LIF) neurons and trained using surrogate gradient and BPTT. E2E is based on a ResNet-8 architecture with IF spiking neurons and is trained using ANN-SNN conversion. Notably, neither networks use threshold balancing to mitigate the quantization error introduced by the spiking neurons. As we can observe from Table 2, Recurrent SNN proposed in Cramer et al. (2022), which features a very simple architecture, is not able to match current state of the art results on this particular task even with a latency of 200 timesteps. At the opposite E2E reaches an accuracy score of 92.9% using 32 timesteps. This accuracy score, which is closer to the state of the art, is nevertheless obtained at the cost of a high latency. At the opposite, our models can consistently outperform recent state of the art SNNs both in terms of accuracy and latency. As an example, we obtain an accuracy of 94.31%, which is only 0.15% below the accuracy of the full-precision ANN using only 4 timesteps. These results confirm the importance of adopting a quantization-aware-training strategy, i.e., direct learning, but also to jointly optimize the spiking neurons parameters, to reduce quantization noise, as our ATIF models do.

4.4. Ablation studies

We decompose the effects of our methods using an ablation study on the CIFAR-10 dataset. Three ResNet-18 networks are trained with different spiking neuron models. The baseline corresponds to a network with IF soft-reset spiking neurons, where all neurons share a fixed $V_{th} = 1$. We compare this network with both uniform and non-uniform quantization schemes described in Sections 3.3 and 3.4, respectively. In those schemes V_{th} is a trainable parameter: it is trained layer-wise for each network. All three networks are trained using surrogate gradient learning and use the same training setup described in Section 4.1.

As shown on Table 3, a significant accuracy improvement can be obtained by optimizing the V_{th} of spiking neurons during training. As an example, comparing the uniform and fixed threshold quantization scheme with the trainable one we can observe that the accuracy increases by 4.16%. We can also observe, that we achieve an accuracy improvement without increasing neither the latency nor the number of the generated spikes. By using a trainable V_{th} scheme, the sparsity can be further reduced, so that we can improve at the same time the performance and the computational efficiency of the network. Finally, using a nonuniform quantization scheme provides a further improvement of 0.81% on the accuracy. However using this scheme, neurons generate slightly more spikes than the uniform quantizer with trainable V_{th} .

5. Discussion and further improvements

In this paper, we have approached the problem of SNNs latency using tools and metrics available within the data compression

and information theory domains. We have shown that the source of accuracy degradation between ANN and SNN resides in the quantization error caused by the discretization of the information exchanged by neurons. By leveraging the techniques used in data compression, i.e., scalar quantization, we show that we can improve the performance of spiking neurons. Moreover, by improving the signal-to-quantization-noise ratio of the neurons we show that we can significantly boost the accuracy of the SNNs overall. Finally, our results are obtained without compromising the efficiency of the resulting SNNs. We are able to approach the performance of full precision ANNs (1.31% difference in the case of ResNet-18 on CIFAR-10 and 0.15% difference on GSC) using only 4 timesteps and 0.5 spikes/neuron on average.

The SNNs presented in this paper are rate-coded networks. Rate coding have long been considered as an inefficient coding scheme, mainly because of the huge number of required timesteps on early works (Sengupta et al., 2019) to approach ANNs accuracies. However, this inefficiency is not intrinsically related to the rate coding mechanism but rather to the quantization scheme used to encode information. Moreover, the methodology that we have used in our analysis can also be applied to other types of coding mechanisms, such as time-coded networks.

While most of the works in the literature still use conversion techniques, our SNNs were trained using a direct learning scheme. With direct learning, time is taken into account during the training process. We can therefore optimize the dynamic behavior of the network. This has some advantages over conversion schemes. For example, Li et al. (2022) identify a source of quantization error, called occasional noise, produced by the oscillation of the input current of the neurons. They reduce the impact of this noise source by introducing a mechanism to generate spikes with negative polarity. This phenomenon is not relevant when the network is trained in the spike domain. Quantization errors caused by the dynamic behavior of the network are taken into account and minimized during the training process. Therefore, we do not need to introduce non-biological plausible mechanisms in our networks.

One of the main drawbacks of the proposed method is the complexity and the memory budget required by the direct learning with surrogate gradient during training. Since each forward and backward pass must be repeated T times to compute the gradients, the training procedure is slower compared to the ANN-SNN conversion where only the ANN is trained, then the weights and biases are transferred to the SNN. Moreover, since direct learning uses back propagation through time (BPTT) it is prone to the vanishing and the exploding gradient problems. If the latter could be mitigated using, for example, gradient clipping the former is more difficult to deal with. With vanishing gradients the learning process of the SNNs slow down. Therefore, to avoid underfit, SNNs training requires a relative large number of epochs and generally a higher learning rate compared to ANNs.

The time dimension is a fundamental part of the SNNs paradigm. SNNs are therefore well-suited to process spatio-temporal information. Therefore, another interesting future direction could be the extension of our work for the case of SNNs processing time varying signals. Our analysis of quantization must be revisited since the input current of the neuron cannot be

considered constant anymore. Finally, we only considered SNNs where weights, biases and neuronal parameters are coded using full-precision representations, e.g., floating point. To obtain full benefits from the low-complexity computational model of the SNNs on neuromorphic hardware the memory footprint of the model must also be considered. In this case it could be possible to integrate a quantization-aware-training procedure, e.g., LSQ, during SNN training to quantize the network parameters using low-precision representations.

Data availability statement

Publicly available datasets were analyzed in this study. This data can be found here: <https://www.cs.toronto.edu/kriz/cifar.html>, https://www.tensorflow.org/datasets/catalog/speech_commands.

Author contributions

AC developed the methods, under the supervision of AP and BM. All authors contributed to the article and approved the submitted version.

References

- Abderrahmane, N., Lemaire, E., and Miramond, B. (2020). Design space exploration of hardware spiking neurons for embedded artificial intelligence. *Neural Netw.* 121, 366–386. doi: 10.1016/j.neunet.2019.09.024
- Chakraborty, A., Panda, S., and Chakrabarti, S. (2022). “Action potential parameters and spiking behavior of cortical neurons: a statistical analysis for designing spiking neural networks,” in *IEEE Transactions on Cognitive and Developmental Systems*.
- Cramer, B., Stradmann, Y., Schemmel, J., and Zenke, F. (2022). The Heidelberg spiking data sets for the systematic evaluation of spiking neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* 33, 2744–2757. doi: 10.1109/TNNLS.2020.3044364
- Diehl, P., and Cook, M. (2015). Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Front. Comput. Neurosci.* 9, 99. doi: 10.3389/fncom.2015.00099
- Diehl, P. U., Neil, D., Binas, J., Cook, M., Liu, S.-C., and Pfeiffer, M. (2015). “Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing,” in *2015 International Joint Conference on Neural Networks (IJCNN)*, 1–8.
- Ding, J., Yu, Z., Tian, Y., and Huang, T. (2021). “Optimal ANN-SNN conversion for fast and accurate inference in deep spiking neural networks,” in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2328–2336 (Montreal, QC: International Joint Conferences on Artificial Intelligence Organization).
- Esser, S. K., McKinstry, J. L., Bablani, D., Appuswamy, R., and Modha, D. S. (2020). Learned step size quantization. *arXiv preprint arXiv:1902.08153*. doi: 10.48550/arXiv.1902.08153
- Fang, W., Chen, Y., Ding, J., Chen, D., Yu, Z., Zhou, H., et al. (2020). *Spikingjelly*. Available online at: <https://github.com/fangwei123456/spikingjelly> (accessed February 06, 2023).
- Fang, W., Yu, Z., Chen, Y., Huang, T., Masquelier, T., and Tian, Y. (2021). “Deep residual learning in spiking neural networks,” in *Advances in Neural Information Processing Systems*, eds A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan. Vol. 34 (Curran Associates, Inc.), 21056–21069. Available online at: <https://openreview.net/forum?id=6OoCDvFV4m>
- Han, B., Srinivasan, G., and Roy, K. (2020). “RMP-SNN: residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Lemaire, E., Cordone, L., Castagnetti, A., Novac, P.-E., Courtois, J., and Miramond, B. (2022). An analytical estimation of spiking neural networks energy efficiency. *arXiv preprint arXiv:2210.13107*. doi: 10.48550/arXiv.2210.13107
- Li, C., Ma, L., and Furber, S. (2022). Quantization framework for fast spiking neural networks. *Front. Neurosci.* 16, 918793. doi: 10.3389/fnins.2022.918793
- Li, Y., Deng, S., Dong, X., Gong, R., and Gu, S. (2021). “A free lunch from ANN: towards efficient, accurate spiking neural networks calibration,” in *Proceedings of the 38th International Conference on Machine Learning*, eds M. Meila and T. Zhang (PMLR), 6316–6325.
- Mendez, J., Bierzynski, K., Cuéllar, M., and Morales, D. P. (2022). Edge intelligence: concepts, architectures, applications and future directions. *ACM Trans. Embedded Comput. Syst.* 21, 1–41. doi: 10.1145/3486674
- Neftci, E. O., Mostafa, H., and Zenke, F. (2019). Surrogate gradient learning in spiking neural networks: bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Process. Mag.* 36, 51–63. doi: 10.1109/MSP.2019.2931595
- Panda, S., Hore, A., Chakraborty, A., and Chakrabarti, S. (2021). “Statistical description of electrophysiological features of neurons across layers of human cortex,” in *2021 Advanced Communication Technologies and Signal Processing (ACTS) (Rourkela: IEEE)*, 1–5.
- Rathi, N., and Roy, K. (2021). DIET-SNN: a low-latency spiking neural network with direct input encoding and leakage and threshold optimization. *IEEE Trans. Neural Netw. Learn. Syst.* 1–9. doi: 10.1109/TNNLS.2021.3111897
- Rueckauer, B., Lungu, I.-A., Hu, Y., Pfeiffer, M., and Liu, S.-C. (2017). Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Front. Neurosci.* 11, 682. doi: 10.3389/fnins.2017.00682
- Sengupta, A., Ye, Y., Wang, R., Liu, C., and Roy, K. (2019). Going deeper in spiking neural networks: VGG and residual architectures. *Front. Neurosci.* 13, 95. doi: 10.3389/fnins.2019.00095
- Srinivasan, G., Panda, P., and Roy, K. (2018). STDP-based unsupervised feature learning using convolution-over-time in spiking neural networks for energy-efficient neuromorphic computing. *J. Emerg. Technol. Comput. Syst.* 14, 44:1–44:12. doi: 10.1145/3266229
- Warden, P. (2018). Speech commands: a dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*. doi: 10.48550/arXiv.1804.03209
- Yang, Q., Liu, Q., and Li, H. (2022). “Deep residual spiking neural network for keyword spotting in low-resource settings,” in *Interspeech 2022 (ISCA)*, 3023–3027.

Funding

This work has been supported by the French government through 3IA Côte d’Azur Institute, reference ANR-19-P3IA-0002.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher’s note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.