IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 3, 2022, accepted May 24, 2022, date of publication June 3, 2022, date of current version June 13, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3179968

Spiking Neural Networks: A Survey

JOÃO D. NUNES[1,2], MARCELO CARVALHO[1], DIOGO CARNEIRO[3], AND JAIME S. CARDOSO[1,2], (Senior Member, IEEE)
[1]Faculty of Engineering, University of Porto, 4200-465 Porto, Portugal
[2]INESC TEC, 4200-465 Porto, Portugal
[3]Bosch Car Multimedia, S.A., 4701-970 Braga, Portugal

Corresponding author: João D. Nunes (jdnunes@fe.up.pt)

This work was supported by European Structural and Investment Funds in the FEDER component, through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) [Project n° 047264; Funding Reference: POCI-01-0247-FEDER-047264].

ABSTRACT The field of Deep Learning (DL) has seen a remarkable series of developments with increasingly accurate and robust algorithms. However, the increase in performance has been accompanied by an increase in the parameters, complexity, and training and inference time of the models, which means that we are rapidly reaching a point where DL may no longer be feasible. On the other hand, some specific applications need to be carefully considered when developing DL models due to hardware limitations or power requirements. In this context, there is a growing interest in efficient DL algorithms, with Spiking Neural Networks (SNNs) being one of the most promising paradigms. Due to the inherent asynchrony and sparseness of spike trains, these types of networks have the potential to reduce power consumption while maintaining relatively good performance. This is attractive for efficient DL and, if successful, could replace traditional Artificial Neural Networks (ANNs) in many applications. However, despite significant progress, the performance of SNNs on benchmark datasets is often lower than that of traditional ANNs. Moreover, due to the non-differentiable nature of their activation functions, it is difficult to train SNNs with direct backpropagation, so appropriate training strategies must be found. Nevertheless, significant efforts have been made to develop competitive models. This survey covers the main ideas behind SNNs and reviews recent trends in learning rules and network architectures, with a particular focus on biologically inspired strategies. It also provides some practical considerations of state-of-the-art SNNs and discusses relevant research opportunities.

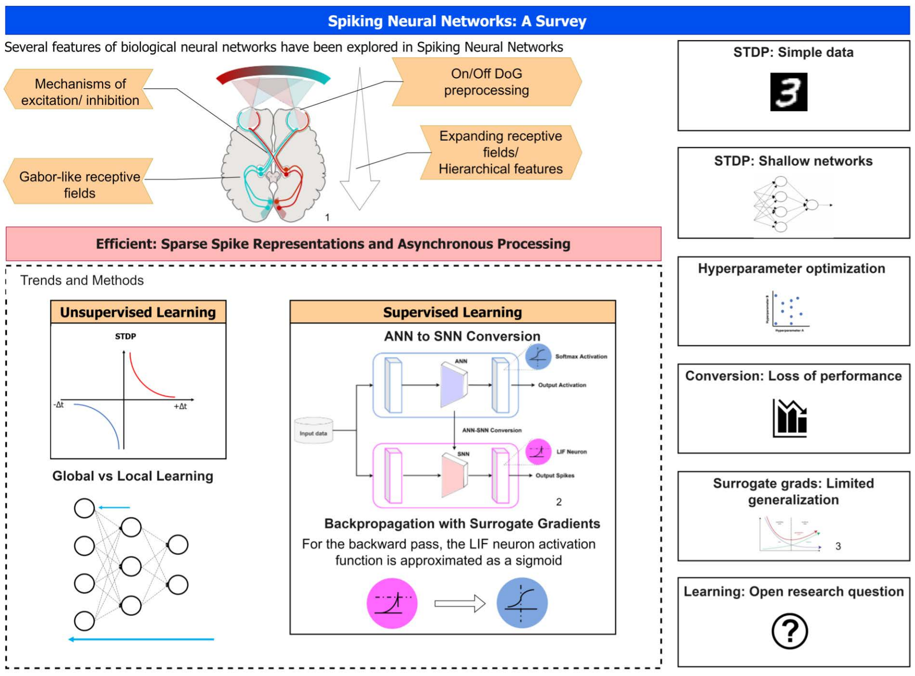
INDEX TERMS Artificial neural networks, computer vision, efficient deep learning, event-driven, machine learning, neuromorphic computing, neuromorphic hardware, spiking neural networks.

I. INTRODUCTION

In the past decade, the field of Deep Learning (DL) has seen a remarkable series of developments, with ever more accurate and robust algorithms that have revolutionized many areas, such as computer vision or natural language processing. Considerable advances in hardware and the introduction of GPU-enabled DL have permitted significant boosts in processing speed and efficiency. Consequently, more complex and robust models have been developed, but the computing power used in cutting-edge algorithms grew even faster.

Often, the starting point of this accelerated growth in DL is marked by the 2012 ImageNet competition, where a team of scientists from the University of Toronto proposed a deep Convolutional Neural Network (CNN), AlexNet [1], an algorithm that had a top-5 error 10.8 % lower than the second-best solution. As a result, deeper networks have been investigated, with an ever-increasing number of parameters and complexity. In the subsequent years, we witness the rapid emergence of several known deep model architectures (e.g., VGGNet [2], ResNet [3], GPipe [4], BERT [5], GPT-3 [6]). However, such outstanding improvements of model capabilities have been correlated with an increase in models' parameters, complexity, prediction latency, training time, etc. Nowadays, training a single top-performing model requires many hardware and energy resources, which results in a large carbon footprint [7]. On the contrary, costs scale at roughly the same rate as the demand in computing power, meaning we are fast reaching a point where DL might become unsustainable [8]. Moreover, some applications might require

The associate editor coordinating the review of this manuscript and approving it for publication was Victor S. Sheng.


Spiking Neural Networks: A Survey
Several features of biological neural networks have been explored in Spiking Neural Networks
Mechanisms of excitation/ inhibition
On/Off DoG preprocessing
Expanding receptive fields/ Hierarchical features
Gabor-like receptive fields
1
Efficient: Sparse Spike Representations and Asynchronous Processing
Trends and Methods
Unsupervised Learning
STDP
-Δt
+Δt
Global vs Local Learning
Supervised Learning
ANN to SNN Conversion
ANN
Softmax Activation
Output Activation
Input data
ANN-SNN Conversion
SNN
LIF Neuron
Output Spikes
2
Backpropagation with Surrogate Gradients
For the backward pass, the LIF neuron activation function is approximated as a sigmoid
STDP: Simple data
STDP: Shallow networks
Hyperparameter optimization
Conversion: Loss of performance
Surrogate grads: Limited generalization
3
Learning: Open research question

1- Source: wikimedia commons. Visual System; Miquel Nieto 2- Source: Adapted from "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, IJCAI-21, Z.-H. Zhou, Ed. International Joint Conferences on Artificial Intelligence Organization, 8 2021, pp. 2328–2336, main Track. [Online]. Available: https://doi.org/10.24963/ijcai.2021/321 3 - Source: S. Mottaghinejad, "The intuition behind bias and variance | Towards Data Science." [Online]. Available: https://towardsdatascience.com/bias-and-variance-but-what-are-they-really-ac539817e171. [Accessed: 30-May-2022].

FIGURE 1. Spiking Neural Networks (SNN) are considered efficient artificial neural network models as they are biologically plausible. However, its activation functions are non-differentiable, meaning backpropagation (BP) cannot be directly employed. Although Spike Timing Dependent Plasticity (STDP), ANN-SNN conversion, and BP with surrogate gradients have been used, the performance is behind conventional ANNs, and the optimum learning strategy is to be found.

thorough consideration of models' parameters and efficiency. A recent survey [9] highlights some key ideas. For instance, specific applications, such as mobile, robotics, or critical systems might require the models to be optimized for the device they will be deployed. Furthermore, special attention must be given when integrating multiple models in the same infrastructure since resources might be exhausted. Therefore, in the face of these challenges, there is a growing interest in developing efficient DL algorithms.

Several strategies can be adopted on multiple levels to approach the challenge of efficient DL. For instance, whereas compression techniques target the representational efficiency of the unified model, learning techniques focus on the training stage. Plus, one could choose to handle the problem at the level of the models' architecture. Nonetheless, a growing paradigm with efficient DL is that of Spiking Neural Networks (SNNs) [10].

Although significant advances have been achieved, the performance of SNNs on benchmark datasets such as MNIST [11], CIFAR-10 [12], or Fashion-MNIST [13] is often lower than conventional Artificial Neural Networks (ANNs). In part, this can be explained by the fact that images on those datasets were acquired resorting to traditional sensors instead of event-driven cameras. However, there are other major drawbacks with SNNs, such as the non-differentiable nature of activation functions due to discrete spike trains, the difficulty in propagating spike information in multilayer unsupervised SNNs, or the local characteristic of biologically inspired learning rules, such as Spike Timing Dependent Plasticity (STDP).

This work covers the main ideas behind SNN models, approaching recent trends in learning rules, network architectures, and biologically inspired strategies (see Figure1). It will also address some practical considerations of state-of-the-art SNNs, further discussing relevant research opportunities. Previous works [14]–[18] have already investigated the main developments in the field of SNNs, but our work differs from these since it is particularly focused on the working principles of biological neural networks. In general, most SNN researches only address a subset of biological mechanisms and different works present distinct strategies, but we argue that maximizing the extent of biological neural network properties exploited in SNNs is a very promising approach and could lead to significant improvements in the performance of SNN models. Moreover, there exist in the literature many promising ideas and conclusions regarding bio-inspired strategies to train SNN models, however, many works are predominantly focused on demonstrating the feasibility and viability of particular components (e.g. learning rules, connection types, etc.) which means there is still the lack of a unifying strategy to develop and train SNN algorithms. Therefore, this work intends to contribute towards bridging that gap by aggregating most of the already existing knowledge regarding SNNs that simulate the working mechanisms of biological neural circuits. To that end, it summarizes the leading ideas and conclusions of recent works, emphasizing the most promising findings and possible future directions in bio-inspired strategies.

Besides this introduction, this work is structured as follows: Section II presents the fundamental principles of SNNs, including neuron models and synapses. Next, section III details the main properties of biological neural networks and summarizes the results of main works that are inspired by the working mechanisms of biological neural networks. Section IV, on the other hand, highlights the main information encoding schemes in biological neuronal networks and how SNNs can mimic those mechanisms further summarizing the relevant findings. In section V we discuss different learning strategies and how the SNN community is addressing the problem. But we also present the results of both the main papers reviewed and of our work in section VI.[1] Finally, we conclude this work and wrap our main findings in section VII.

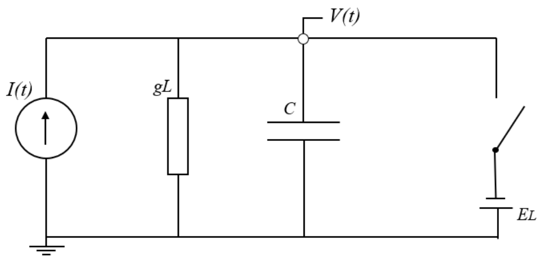
II. FUNDAMENTALS OF SPIKING NEURAL NETWORKS

A SNN architecture consists of neurons interconnected by synapses that determine how information is propagated from a presynaptic (source) to a postsynaptic (target) neuron. The activity of the presynaptic neurons modulates the activation of the corresponding postsynaptic neurons and, unlike the conventional ANNs, the information in SNNs is encoded and transmitted in the form of spikes. Each input is presented for a prespecified amount of time (T), meaning that instead of a single forward propagation, SNNs typically have multiple, $\frac{T}{\delta t}$, forward passes. But as in its biological counterpart, once a presynaptic neuron activates, it sends a signal to its postsynaptic equivalent, in the form of a synaptic current, that is proportional to the weight, or conductance, of the synapse.

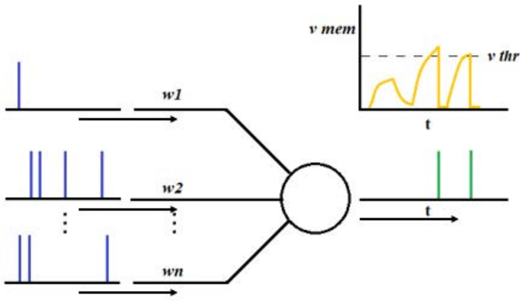
In general, as the synaptic current reaches the target neuron it will alter its membrane potential (v_{mem}) by a certain amount, δv. If v_{mem} reaches a predefined threshold (v_{thresh}), the postsynaptic neuron will emit a spike and reset its membrane voltage to the resting potential (v_{rest}). Notwithstanding, many strategies can be adopted to model the neuron and synapse dynamics. On top of that, different network architectures and applications might require specific combinations of learning rule, v_{mem} dynamics, and neuron model.

There is a broad pool of neuron models and often the literature reports networks that establish the Mcculloch-Pitts, [19], Izhikevich, [20], CSRM or SRM0 [10] neuron models as its basic units. However, the Leaky Integrate and Fire (LIF) and its variants is one of the most popular. In LIF models, the neuron is modelled as a parallel Resistor-Capacitor (RC) circuit with a "leaky" resistor [21], as represented in Figure 2a. The output voltage $V(t)$ of this circuit, (analogous to v_{mem}) is then

[1]Source code available at: https://github.com/joao-nunes/spiking-neural-networks-a-survey.git



(a) LIF neuron model as a parallel RC circuit.



(b) Evolution of v_{mem} in a LIF neuron for a set of input spike trains.

FIGURE 2. If there is no input current, $V(t)$ will decay to its resting potential E_L, otherwise it will increase by δv, as established by Equation 2. The capacitor will discharge (comparable to a neuron emitting a spike) when $V(t)$ reaches the predefined threshold. The synaptic conductance is here represented by the W vector. It is possible to observe that if the input is not sufficient, the neuron will not fire. Also, it is observable that in the first moments after the neuron has emitted a spike (t_{refr}), it cannot fire again, regardless of the input it is receiving.

mathematically defined as:

$$C\frac{dV}{dt} = -g_L\left(V(t) - E_L\right) + I(t) \tag{1}$$

From 1, we see that $V(t)$ is dependent on the conductance, g_L, of the resistor, the capacitance, C, of the capacitor, on the resting voltage (E_L) and of a current source $I(t)$. If we multiply 1 by $R := \frac{1}{C}$, we obtain $\frac{dv_{mem}}{d_t}$ in terms of the membrane time constant, τ_m:

$$\tau_m\frac{dv_{mem}}{d_t} = -\left[v_{mem}(t) - v_{rest}\right] + RI(t) \tag{2}$$

We observe that, due to the leaky behaviour of the model, v_{mem} is constantly decaying to its rest value. Another important consideration for LIF-neurons is that it can endure a refractory period, i.e., a period after reset during which the neuron cannot fire again, regardless of its input. Figure 2b illustrates the behaviour of a LIF node on spike train input.

The activation function, $A(t)$, of LIF neurons is thus defined as:

$$A(t) = \begin{cases} 0, & \text{if } v_{mem} < v_{thresh} \\ 1, & \text{if } v_{mem} \geq v_{thresh} \end{cases} \tag{3}$$

A major drawback of SNNs is the non-differentiable nature of its activation function 3, meaning that Backpropagation (BP), the most widely used learning algorithm in ANNs, cannot be directly employed [22]. But for the network to be able to learn, we must decide proper strategies to update the synaptic weights, W. One of the prevailing methods is the biologically inspired STDP. STDP results from a set of neurobiological findings that started in 1949, with Donald Hebb, who proposed a fundamental principle of synaptic plasticity to describe how learning might be accomplished in the brain. In "The Organization of Behaviour: A Neuropsychological Theory" [23] he famously states: "When an axon of Cell A is near enough to excite a Cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased." Later, his work was supported with the discovery of Long Term Potentiation (LTP) [24], a neural mechanism that describes a persistent increase in synaptic strength if a presynaptic neuron fires briefly before its postsynaptic equivalent [25]. On the contrary, Long Term Depression (LTD) refers to the process of decreasing the synaptic effectiveness of a presynaptic neuron when there is no evident causal relationship between its spiking activity and that of the postsynaptic neuron [25]. More specifically, STDP is an asymmetric form of Hebbian learning rules and it establishes that the synaptic weight change is proportional to the relative timing between pre-and postsynaptic activations [26]. Equation 4 formally describes this mechanism:

$$\Delta w = \begin{cases} Ae^{\frac{-(|t_{pre}-t_{post}|)}{\tau}}, & \text{if } t_{pre} - t_{post} \leq 0 \\ Be^{\frac{-(|t_{pre}-t_{post}|)}{\tau}}, & \text{if } t_{pre} - t_{post} > 0 \end{cases} \quad (4)$$

where $A > 0$ and $B < 0$ define the learning rates, τ is the timing window constant, t_{pre} and t_{post} are the absolute timings of pre- and postsynaptic spikes and Δw is the synaptic weight update. From this system of equations, it is observable that the first branch refers to LTP and the second branch to LTD. Notably, STDP is a local learning rule, meaning it does not consider global information for weight updating and, as this work will discuss later, it becomes challenging to train SNNs with direct BP.

III. FROM THE RETINA TO THE VISUAL CORTEX: BIOLOGICALLY INSPIRED SPIKING NEURAL NETWORKS

ANNs are inspired by the mammalian brain, yet their principles are fundamentally different, more specifically in what refers to their structure, learning rules, and computations. For instance, whereas ANNs approximate neurons as non-linear continuous functions, biological neurons compute asynchronous event streams through discrete and temporally precise action potentials [15]. Although ANNs (e.g., CNNs, Recurrent Neural Networks (RNNs), Transformers, etc.) have achieved outstanding results in many Machine Learning (ML) tasks, they are incredibly inefficient in comparison with biological neural circuits. SNNs, on the other hand, exhibit many properties in common with biological neural networks, namely sparsity of computations, low power consumption, fast inference, or a considerable degree of parallelism [15]. For this reason, there is a growing effort from neuroscientists to better understand the principles behind learning and information processing in the mammalian brain. Notably, one of the most widely understood brain functions is the visual system and the Primary Visual Area (V1). Many correlates of human vision have thus served as inspiration for several SNN strategies and their applications to computer vision tasks.

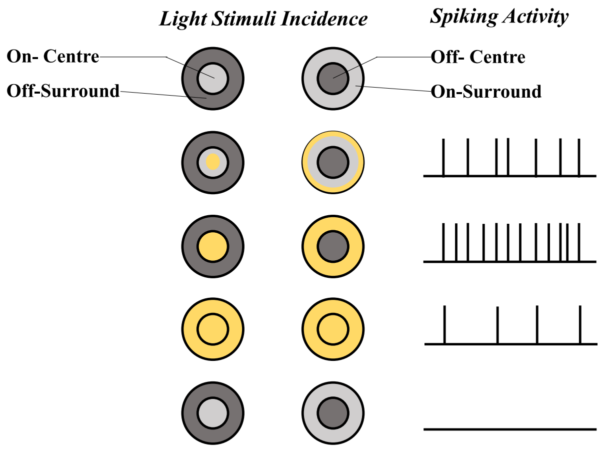
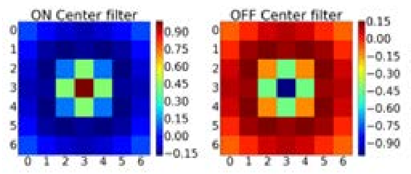


FIGURE 3. On-centre- and Off-centre-surround bipolar cells' response to light stimuli. If there is maximum overlap between light incidence and the cell's RF, it outputs a maximum response pattern, demonstrated by the highest firing rates. However, if there is a weak correspondence, we observe a weak firing response or no spikes at all.

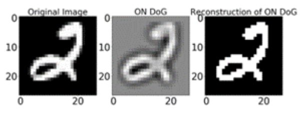
A. THE MAMMALIAN RETINA: INPUT STIMULI SELECTIVITY

As stated previously, SNNs are promising for event-driven computations. This is primarily encouraged by the biological structure and working mechanisms of the mammalian retina. The mammalian retina is a multilayer system, containing thousands of photoreceptors. Photoreceptors act as transducers, converting light signals to discrete electric signals (spikes) which are further processed by the brain. In turn, the photoreceptors connect to ON-centre and OFF-centre bipolar cells. ON-centre cells are depolarized by stimulating the centre of the Receptive Field (RF), whereas OFF-centre cells are depolarized by lack of stimuli at the centre of their respective RF. Moreover, ON/OFF bipolar cells possess a surrounding region in their RFs with inverse properties [27]. Figure 3 illustrates the response of ON-centre- and OFF-centre-surround cells to different light inputs.

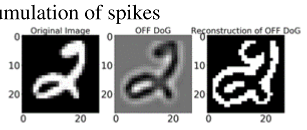
These mechanisms demonstrate the retina's role in preprocessing input stimuli and suggest some form of selectivity. ON-centre/OFF-centre bipolar cells are responsible for coding contrasts in luminance, meaning that they ignore redundant information and instead focus on relevant intensity changes (events). Next, bipolar cells connect to ganglion cells, the output neurons of the retina. Evidence supports that ganglion cells are selective to a plethora of input stimuli features, including colour, luminance contrasts, object size, and stimulus direction and orientation [28]. Notably, at the retina level, the RFs present a more basic structure and encode



(**a**) On/Off DoG filters. On-centre filter has higher values in the centre as opposed to the Off-centre filter. Colour code shows the filter values.



(**b**) Left: Original gray-scale image. Centre: Output of the On DoG filter. Right: Accumulation of spikes



(**c**) Left: Original gray-scale image. Centre: Output of the Off DoG filter. Right: Accumulation of spikes

FIGURE 4. **On-centre/Off-centre DoG filters. Source: [31].**

a broad range of spatial frequencies [29]. Many of such properties have already been explored in the development of bioinspired SNNs. For example, the work of [30] proposes a 3 stages SNN architecture to process Dynamic Vision Sensor (DVS) data. Their algorithm contains an Unsupervised Learning (UL) layer for spatio-temporal pattern extraction, that mimics ganglion cell selectivity, and a Supervised Learning (SL) layer inspired in the retinotopic organization of the visual cortex to classify the MNIST-DVS dataset. In turn, [31] apply ON-OFF Difference of Gaussian (DoG) filtering, before encoding the input images into spike trains and feeding them to a convolutional SNN classifier. Figure 4 illustrates the obtained DoG filters as well as 2 example outputs, after convolution with On- and Off-DoG filters.

Similarly, [32] use ON-centre and Off-centre DoG convolutions in a hierarchical multilayer SNN for corner detection and, recently, [33] proposed the use of on-centred input neurons with a 5×5 receptive field to mimic how ganglion cells perceive the visual scene. But despite these developments, it remains challenging to model the behaviour of ganglion cells in ML models, partly due to their diversity, apparent fine-tuning for specific features of the visual scene, distinct functions, subtypes, and connections to downstream visual processing pathways [34], [35]. A possible solution could be to model the retina using CNNs. Indeed, there is a certain degree of realism in this approach since the eyes contain very few Feedback (FB) connections, meaning Feedforward (FF) networks, like CNNs can be used [36]. Nevertheless, modelling the behaviour of the retina and ganglion cells seems to be a promising avenue for the development of high-performing and efficient SNNs. More concretely, by mimicking the ability of these cells to encode the visual scene into a potentially abstract subset of features, multilayer hierarchical SNNs models could better discriminate between classes by taking advantage of more relevant and complementary information.

B. THE NEURAL CIRCUITS: INFORMATION TRANSMISSION

Ganglion cells will subsequently send output spikes to visual processing centres in the brain. Information is thus propagated and processed through an intricate network of neurons that form complex circuits. Notwithstanding, these networks can be decomposed into much simpler circuits that perform elementary operations. The most widely used neural circuit is the FF excitation (Figure 5), and although simple, it permits the flow of information through the network. At each layer, every neuron receives input from multiple presynaptic nodes through converging connections and is itself divergently connected to multiple postsynaptic equivalents. Another advantage of this circuit is it can increase the Signal to Noise Ratio (SNR) considering that different neurons process similar inputs but uncorrelated noise [37]. Although FF excitatory circuits fulfil a prominent role in propagating information to deeper brain regions, other mechanisms occur locally that regulate the relationship between input stimulation and output spiking activity. Two of said mechanisms are FF and FB inhibition circuits.

In FB inhibition, a layer of presynaptic excitatory neurons will stimulate postsynaptic excitatory neurons as well as the inhibitory neurons that project back to the presynaptic excitatory layer. In turn, FF inhibition refers to the case where the postsynaptic population of inhibitory neurons connects to postsynaptic neuron populations. It is believed that FF/FB inhibition act dynamically to perform a plethora of functions. These include: controlling of synchronous or oscillatory spiking activity, regulation of neuron sensitivity to input stimuli, through the promotion of a fast response, and preventing quiescence or saturation of postsynaptic firing rates [38]–[42]. Lateral inhibition circuits also partake in neural networks. It consists of the capacity that some neurons present to reduce the activity of parallel pathways. Through this, it is possible to exacerbate some activity whilst reducing the transmission of less relevant spikes. Lateral inhibitory circuits enhance contrasts and carry out a significant role in decorrelating neural responses, therefore, promoting the discrimination of similar stimuli [43]. For example, it is believed to occur at the retina, where active photoreceptors, through horizontal cells, will inhibit neighbouring photoreceptors [37].

The work of [44] has demonstrated positive results when using FB inhibition to promote the separability of similar input stimuli. The increase in selectivity is explained by the suppression of non-informative activity. But there are a few considerations with this approach. The solution is supported on the basis that FF connections from uninformative neurons, coding stimuli overlap, will dominate the discriminative connections. Beyond a certain degree of overlap, it can even prevent the informative output neurons from firing at all. To overcome such limitations, a mechanism of dynamic excitation/inhibition was proposed, where FB

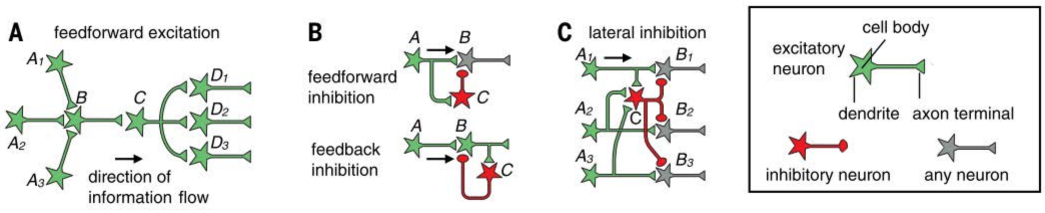


FIGURE 5. Neural circuits of excitation and inhibition A: FF Excitation. B: FF and FB inhibition. C: Lateral inhibition. Source: [37].

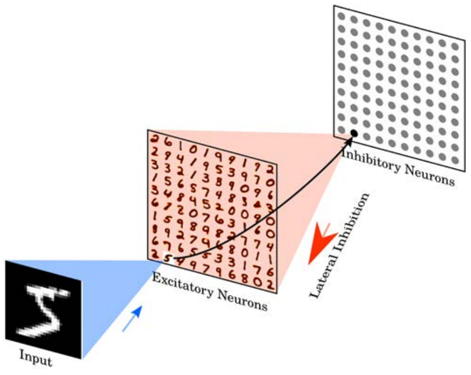


FIGURE 6. Unsupervised SNN architecture proposed by [45]. It shows the input connections to an example excitatory neuron. Excitatory neurons are connected to inhibitory neurons via one-to-one connections, as shown for the example neuron. The red shaded area denotes all connections from one inhibitory neuron to the excitatory neurons. Each inhibitory neuron is then connected to all excitatory neurons, except for the one it receives a spike from. Source: [45].

inhibitory connections were strengthened (with anti-Hebbian learning) every time excitatory connections were potentiated. The results show this strategy promotes the learning of selective responses and increases the discrimination of similar input patterns. This is a biologically realistic approach, since neural circuits of multilayer excitation/inhibition could partake in the suppression of uninformative stimuli. Remarkably, the authors suggest this strategy could be incorporated in hierarchical multilayer networks, with FB inhibition from higher to lower layers, to allow the recognition of more complex patterns.

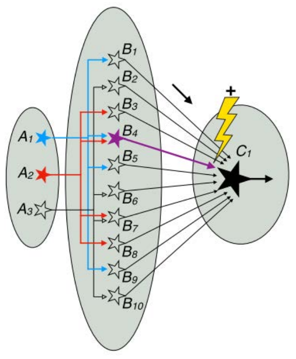
All things considered, inhibition regulates spatio-temporal dynamics of fundamental importance to determine the performance of neural networks and specifically in the context of SNNs, it has already been explored. For instance, [45] use lateral inhibition to promote competition between neurons. With this, they propose a model where each neuron's RF, encoded in its synaptic weights, will correspond to a single data sample or the average of a limited subset of data samples. Then, when a single input is presented, the firing response of each prototype is used to predict the corresponding class. Figure 6 illustrates the presented network.

[33] on the other hand, propose an ensemble of hierarchical unsupervised SNNs. They interactively combine excitatory and inhibitory neurons. This strategy allows regulating network states and promotes distinguishable activations. Considerably, they add lateral inhibition to perform the classification task. [46], in turn, argue that dynamic excitatory and inhibitory neural circuits facilitate convergence and improve the performance of neural networks. The authors propose a deep SNN that resorts to the mechanisms of adaptive self-FB and balanced excitatory and inhibitory neuron circuits, whilst suggesting such strategies could accelerate the training of SNNs. Another work [47] demonstrates the potential of global FB connections and local learning rules for multilayer SNNs. They train the network in a 2-step approach. First, a FF pass to obtain the predictions, then FB is used to obtain the targets of the various hidden layers. In the second step, the loss is computed at the last layer, and the prediction error is propagated to hidden layers, through global FB connections. The weights of hidden layers are thus updated locally, resorting to STDP. In this way, the proposed architecture solves the BP challenges of SNNs, whilst avoiding transmitting the error layer by layer. FB connections are equally suggested by [48]. They introduce a novel way of training FB SNNs, based on the implicit differentiation on the equilibrium state. In this work, it is suggested a certain similarity to Hebbian learning, but the weight updating strategy considers both average firing rates and temporal information.

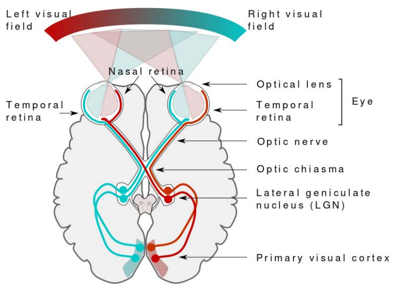
Despite the aforementioned contributions, there exist other relevant circuit designs that could potentially lead to performance gains and increased versatility of SNN. Often, traditional ANNs perform several steps of dimensionality expansion (feature generation) and dimensionality reduction (feature selection). This is largely inspired by the neural circuits of living organisms. Furthermore, it is firmly established that dimensionality expansion increases the separability of neural representations. More concretely, lower-dimensional representations produce distinct responses to specific features, whereas expansion allows the representation of said separable representations as well as their conjunction, thus augmenting the expressive basis and subsequent linear separability of classes [49]. For instance, a study [50] highlights that sparse expansion implemented by FF connections increases robustness to the variability of the input signal, i.e., increases de SNR. Furthermore, it can also decrease the dissimilarity between classes by encoding input statistics into the expanding connections' synaptic weights. Besides, increasing the dimensionality of the inputs also augments the performance of the classifier [50]. However, the separability achieved through dimensionality expansion comes at the cost of generalizability. Meaning, it increases the sensitivity to noise [49]. But to overcome such limitations, compression can reduce the dimensionality and extract meaningful representations from the data. Put differently, there is a need of achieving a balance between expansion and compression, depending on the task objectives. Figure 7a illustrates a simplified neural circuit of dimensionality expansion and reduction. There are many well-known examples of ANNs that have defined dynamic pathways of expansion and compression (e.g., RNNs [51]). Similarly in SNNs, a few works also revolve around these concepts. For instance, [52] propose reservoir computing techniques, namely Liquid State Machines (LSMs), and optimization strategies to develop networks suitable for neuromorphic computing. With this learning approach they are able to find an optimum SNN network topology (i.e., connections) and corresponding hyperparameters (e.g. number of neurons, leakage constants, etc.) that can serve as the *Liquid* of a LSM. This model thus incorporates the dynamic (time-varying) behavior of recurrent SNNs. The readout layer is then trained to classify information from the extracted reservoir state vectors. [53], on the other hand, propose deep LSMs with randomly initialized hidden layers, interleaved with Winner-Take-All (WTA) layers, trained with STDP, to achieve a low-dimensional representation of the information captured by high-dimensional hidden layers. The authors argue this leads to hidden layers capable of processing information over multiple time-scales. The hidden neurons are set to 10x the dimension of the input space and consist of primary neurons, connected to the input layer, and auxiliary neurons which only have recurrent connections within the hidden layer. An attention-modulated readout layer is then stacked on top of the *Liquid* to perform classification on the DogCentric [54] video recognition dataset. In turn, [55] suggest the readout layer of a LSM to be trained with Backpropagation Through Time (BPTT). In their work, the authors propose the *Liquid* to have recurrent connections and static weights, randomly initialized. They then transfer their GPU-trained network to neuromorphic hardware, suggesting competitive performance on the N-MNIST [56] dataset. Nonetheless, we argue that despite challenging, leveraging these strategies in SNNs could lead to performance gains by augmenting the expressive basis of neuron populations.

C. THE VISUAL CORTEX: INFORMATION PROCESSING

We have seen that light stimuli will be pre-processed at the retina and then propagated through an intricate network of several simple circuits. But it remains to discuss how information is processed at the different building blocks of the visual pathway. Indeed, it is supposed that the mammalian brain processes visual information in a bottom-up approach, where each subsequent layer is more specialized than the previous. The goal is to extract salient points, or groupings, of low-level features that can reduce redundancy and complexity of the scene but encode relevant information. It all starts at the retina, where ON- and OFF-centre- surround



(a) Schematic of dimensionality expansion and reduction in neural circuits. Source: [37]



(b) Visual pathway of the human brain. The primary processing centres are the retina and V1. [2]

FIGURE 7. Information is propagated from the retina to the LGN through the optical nerve and then to the visual cortex.

cells will impose some form of pre-processing to the input stimuli. At this stage, more than a dozen different types of ganglion cells will split visual information into separate input streams, representing unique low-level features of the visual scene. Although not yet fully understood, it is supported that these features entail, among others, luminance contrasts, direction selectivity, edge selectivity, motion sensitivity, and colour [34], [57]. Next, the axons of the retinal ganglion cells form the optical nerve and project towards the Lateral Geniculate Nucleus (LGN). From here, visual information travels to the visual cortex. Figure 7b demonstrates the visual pathway of the human brain.

As information propagates from the retina to LGN and through the visual cortex, neurons' RFs become more complex, covering ever bigger regions [58]. More precisely, it was suggested that these RFs are organized according to their complexity. Neurons with simple RFs, respond to stimuli of specific slit width, slant, orientation, and position. Complex RFs on the other hand, cover a wider range of the visual scene instead of a single position but equally respond to slit-shaped stimuli. In turn, neurons with hypercomplex and higher-order hypercomplex RFs require elaborate inputs to activate [59]. The visual cortex, located at the occipital lobe, is the primary unit responsible for combining the elementary features incoming from the LGN so that we can build a complete

[2]Source: wikimedia commons. Visual System; Miquel Nieto

representation and understanding of the visual scene. V1, in the visual cortex, is understood as the first stage of cortical processing. V1 neurons are, thus, categorized into simple, complex, and hypercomplex cells. Simple cells are perceived to compute linear combinations of incoming streams of information, whilst complex cells perform operations on the output of simple neurons [60]. Notably, simple neurons can also be categorized into ON-centre and OFF-centre cells. Complex neurons, on the other hand, will produce a sole response. Furthermore, it is well established that neurons in V1 are organized in a retinotopic manner, meaning neighbouring neurons represent neighbouring positions in the visual field and that its RFs completely represent the visual scene.

Nonetheless, these neurons present some form of selectivity for a panoply of characteristics, similarly to retina ganglion cells. However, V1 neurons are more selective than the former. Two of the major attributes of V1 neurons are their selectivity to orientation and spatial frequency. More precisely, V1 neurons operate at various scales, with little contextual information, and the way it performs salient point detection strikingly resembles 2D Gabor functions [61], [62]. 2D Gabor functions are defined as complex sinusoids, modelled by a Gaussian envelope. Equation 5 defines the general case, for spatial coordinates:

$$G(X, Y) = \mathrm{e}^{-\frac{X^2+\gamma^2 Y^2}{2\sigma^2}} \cos\left(\frac{2\pi}{\lambda} X\right) \tag{5}$$

With γ the aspect ratio, λ the wavelength of the sinusoid, and σ the effective width. Due to their nature, Gabor functions are sensitive to edges, orientation, and texture, operating at different frequencies and scales [63], just like V1 neurons' computations. In this way, V1 neurons encode information in a sparse basis set that represents the entire visual scene. This encoding proves efficient and attractive for the development of SNNs since a linear combination of a limited number of neurons' RFs can be used to reconstruct the whole visual input. That is, only a few neurons are required to be active at a time [64].

V1 output will then stimulate the extrastriate visual areas, V2 and V4, and afterwards, visual information is sent to the Infrotemporal (IT) cortex [39]. At each stage, neuron populations will combine stimuli from previous layers, therefore forming progressively larger and more complex RFs. From simple oriented bars and edges at V1 to moderately complex features, like corners, in V2 and V4, to complex objects and faces, at the IT cortex [65]. Another relevant property that arises from the aforementioned visual processing pathway is invariance to stimuli position and scale, as RFs become larger. Figure 8 summarizes the progression of input stimuli from the retina to the visual cortex.

Although CNNs are not biologically plausible, they are indeed inspired by and present many similarities with biological neural networks of the mammalian's brain. For example, CNNs can also present the properties of invariance to scale, position and slight deformities of the input stimuli [66]–[70]. Moreover, the first layers of CNNs trained on natural images have been shown to present Gabor-like RFs [1]. Besides, most CNNs are also organized hierarchically, with deeper layers progressively encoding more abstract and complex features.

Similarly, some SNN architectures also simulate the visual information processing of the ventral pathway. For instance, [71] used a biologically plausible multilayer hierarchical SNN to classify the benchmarking MNIST [11] and CIFAR-10 [12] datasets. The proposed architecture consists of 6 layers: an encoding layer to simulate the retina and 2 convolution layers that, together with 3 pooling layers, emulate the visual cortex. Promisingly, their work incorporates many known properties of the visual pathway. It includes shift-invariance, edge-like filters at the first convolutional layer (V1), and increasing RFs. In the same way, [33] suggest an ensemble of hierarchical multilayer SNNs, where each model is composed of several convolutional and pooling layers, simulating the working mechanisms of the primate brain. Other authors have proposed spiking convolutional neural networks, inspired in V1 computations. Whereas several works propose hand-crafted kernels of fixed parameters [72]–[76] suggest a hierarchical unsupervised algorithm, based on STDP and spike time coding to classify several datasets. The network is comprised of an encoding layer followed by a sequence of several convolutional and pooling layers. Besides the biologically inspired hierarchical structure, the proposed solution is also translation invariant, due to pooling operations. Furthermore, to mimic the retina ganglion cells, the first layer uses DoGs filters, to detect contrasts. These filters are then encoded in spike latency. In turn, [77], despite using a shallow SNN, show that localized Gabor-like receptive fields in conjunction with unsupervised STDP offer a promising solution to increase the performance of SNNs. Moreover, the authors argue that biologically plausible DL demonstrates great potential to improve the performance of SNNs.

We have seen that SNNs are biologically plausible. Nonetheless, their performance is behind that of CNNs, and more work is needed to bridge the gap. Although existing SNNs might be inspired by some of the operating principles and properties of the mammalian brain, more mechanisms could be explored. Markedly, many of the proposed algorithms are shallow and do not explore to the full extent the bottom-up strategy characteristic of visual processing in the mammalian brain. With this in mind, we suggest that integrating more of known biological mechanisms into SNNs architectures could lead to performance gains. In concrete, we identify that concepts such as neural circuits (e.g., dynamic inhibition/excitation, dimensionality expansion, etc.), and ON-center/OFF-center surround RFs, or known properties of the visual processing pathway (e.g., V1 Gabor-like selectivity, hierarchical structure, etc.) need to be more extensively studied and integrated with SNNs to allow the development of energy and data-efficient DL and that can compete with state-of-the-art algorithms like CNNs.

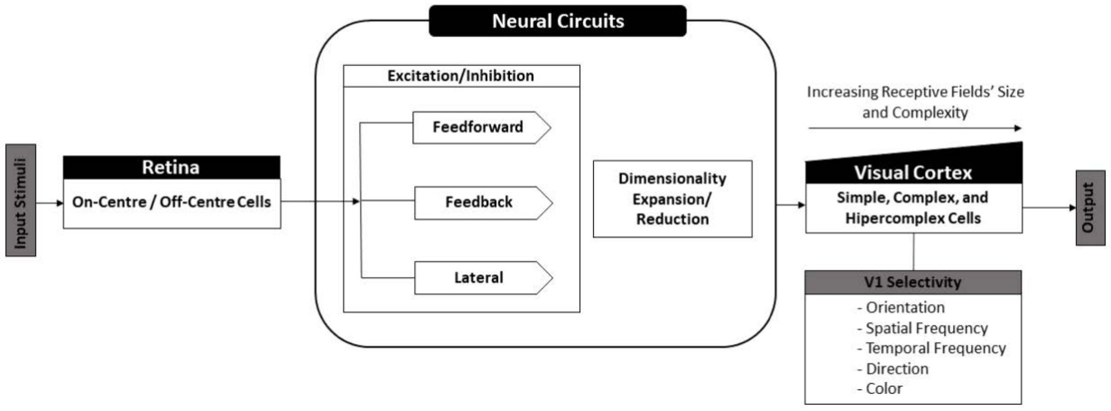


FIGURE 8. Summary of processing of visual information in the mammalian brain. Information travels from the retina to the visual cortex through an intricate network of neural circuits forming, at each stage of the visual processing pathway, larger and more complex RFs. V1 neurons are supported to present Gabor-like RFs.

IV. INFORMATION ENCODING

As mentioned previously, SNNs require the encoding of information, which constitutes a considerable difference compared to traditional ANNs. In fact, in image-based tasks, traditional ANNs use pixel intensities to extract critical features, while in SNNs, there is the need to map each pixel intensity to a discrete spike domain before feature extraction.

The mammalians' brain is extraordinarily efficient when performing complex tasks, and the goal of SNN architectures would be to mimic as close as possible that organ's properties, thus becoming resource-efficient and suitable for applications with certain hardware constraints. Henceforth, the employment of biologically plausible information encoding methods has a crucial role in SNN-based architectures. In fact, the mammalian brain is remarkable in the sense that it performs task-specific encoding, suggesting that ML practitioners could also select different encoding schemes, depending on the application.

There are several encoding strategies, but broadly speaking, these can be divided into two critical groups: rate and temporal coding. As the names suggest, rate coding is based on spikes' firing rates to represent information whilst temporal coding considers the spike times, thus, in general, allowing faster responses. Despite for many years being thought that the mammalian brain exercised, essentially, rate coding, evidence supports the idea that it must also rely upon temporal encoding strategies.

Rate coding schemes are often categorized into count-, density- or population rate methods (Figure 9), as suggested by [78]. In count rate coding, also defined as frequency coding, the mean firing rate (υ) is computed over a prespecified time window (T), as formally defined by Equation 6.

$$\upsilon = \frac{N_{spike}}{T}, \tag{6}$$

with N_{spike} being the number of spikes in a stimulus time window, T. This is the most common rate coding scheme, and it converts each pixel intensity to a spike train. Typically, the probability of spike occurrence at a given time instant, t, is modelled by a Poisson distribution in which the mean event rate is proportional to the pixel intensity. Several studies evidence the existence of biological rate coding. [79] performed the first known experiment that allowed to verify the existence of a biological rate coding. They submitted a frog's muscle to different weights and, through the measurement of nerve action potentials using a capillary electrometer, concluded that the firing rate was proportional to the force exerted by the weights. [80] also describe the crucial role that rate coding plays in human voluntary muscle contraction, mostly for intermediate and higher forces. Furthermore, some experiments corroborated the Poisson-based coding hypotheses through the analysis of biological recordings of the medial temporal and primary visual cortices of macaque monkeys [81].

Density rate coding, on the contrary, represents a non-biologically plausible method that averages the neural activity over several simulations. The spike density, *p(t)*, is therefore defined as

$$p(t) = \frac{1}{\Delta t}\frac{N_{spike}(t; t + \Delta t)}{K} \tag{7}$$

where, first, the number of spikes, N_{spike}, is averaged over a specified time interval [t; t + Δt], that defines the duration of the simulation and, second, the average firing rate is also averaged over K simulations.This is also known as Post-Stimulus Time Histogram (PSTH). The main problem with this approach is that it is impossible for biological neural networks. Input stimuli must be processed in a single run. Nonetheless, averaging a neuron's response over simulations allows the smoothing of intrinsic and network-related noise spikes that occur both *in vivo* and in simulated neurons [82].

Another way of using the mean firing rates to encode information is through the population rate coding scheme, which is in many ways similar to density coding, except that it averages over several neurons instead of over simulations.

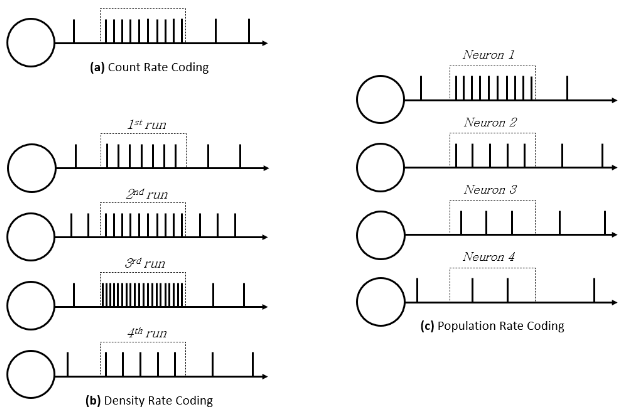


FIGURE 9. Schematic representation of the discussed rate coding methods, where the dotted window represents the stimulus. Source: Adapted from [78].

In this technique, the firing rate *A(t)* is obtained by averaging the number of spikes N_{spike} in the time interval [t; t + Δt] over the number of neurons N and the duration Δt, as described in 8.

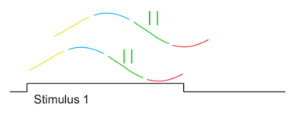
$$A(t) = \frac{1}{\Delta t}\frac{N_{spike}(t; t + \Delta t)}{N} \tag{8}$$

[83] assessed the N1 response through magnetoencephalography by presenting two unique sounds coming from different locations, referenced as adaptor and probe locations. N1 attenuation reflects the degree of overlapping of the neurons responding to the adaptor and the probe sounds. The participants of the study were initially exposed to the adaptor sound and, after some defined time interval, the probe sound and the N1 attenuation were measured. It was verified that the increase of the spatial separation between the adaptor and the probe led to a decrease in attenuation, confirming the existence of a population rate coding method in the human auditory cortex. More recently, [84] tested the rate population coding hypothesis against the labelled-line one to attempt to explain how the sound direction is perceived and if it depends on sound intensity. In this experiment, individuals were submitted to various sound intensities, and interaural time differences, and asked to indicate the perceived laterality. The labelled-line model states that each receptor only responds to a certain stimulus, implying that the response should be intensity invariant while in the population rate coding model each receptor responds to several stimuli, depending on the response of a population of neurons. The obtained results validate the population rate coding hypothesis, since it was verified a midline bias to the perceived laterality for lower sound intensities. The authors point out that, since human visual perception is also based on interocular disparity, the brain may operate a similar coding method in visual perception tasks. Some evidence shows the presence of population rate coding in both macaque and mouse visual systems [85], [86].

Although rate coding methods are widely used in SNNs, the need to average over a certain time interval compromises the responsiveness of the systems since it requires a sufficiently large time window to increase its performance. On the contrary, biological systems need to respond almost instantly to stimuli, which is only possible through methods that implement precise timing instead of mean firing rates. Biological evidence of these methods was provided by [87] that, through the measurement of event-related potentials, concluded that the human brain needs less than 150 ms to process a complex visual task. In the experiment, an image was flashed for 20 ms where subjects had to decide the presence or absence of an animal in the stimulus. This experiment paved the way for the study of more temporally precise encoding strategies, with temporal coding being now extensively studied for computer vision applications. From the available techniques, we highlight Time-To-First-Spike (TTFS), Rank Order Coding (ROC), phase coding, and burst coding methods due to their wider adoption.

TTFS is the simplest case of temporal coding, and it considers the precise time window between the beginning of a stimulus and the first spike emitted by a neuron. With this strategy, information is encoded such that signals with high amplitude trigger an early response, while low amplitude signals translate to late spikes or no spikes at all. [88] found strong evidence of a correlation between the latency of the first spike and stimulus contrast in the retinal pathway. It also demonstrated that spike count affects the response to the said stimulus. In the inferior visual cortex, experiments showed that the first spike contains more information than the combined spikes [89]. In turn, [90] demonstrated the importance of first spikes in visual processing tasks. By submitting mice to trivial discriminating visual tasks and silencing their primary visual cortex, in well-defined intervals, after stimulus onset, they concluded that most neurons emitted as little as one spike or no spikes at all. Besides, only 16% of the neurons spiked more than twice.

Similarly to TTFS, rank order coding (ROC) considers the relative timing of spikes, but across neuron populations. Naturally, the amount of information that can be encoded is constrained by the size of the neuronal population. Biological evidence of this method was presented by [91]. Through retinal recordings, using multielectrode arrays, they analyzed the response of mice Retinal Ganglion Cells (RGCs) to several stimuli. Their work concludes that a single pair of those cells did not provide sufficient information, meaning a larger population of RGCs would be necessary to encode all of the information. In turn, phase coding considers a global reference in the form of a periodic oscillatory signal, where spike times are phase-locked in relation to that reference oscillation (Figure 10). There is also biological evidence of this encoding strategy. For instance, [92] recorded local field potentials in macaques' V1 while presenting a colourized movie and demonstrated that both the spike count and low-frequency local field potentials provided important information regarding the film. A similar study [93] showed that



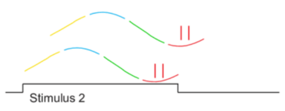


FIGURE 10. Schematic representation of phase coding. In this case, it is not possible to distinguish each stimuli through their spike count (since it is the same for both) but rather through the timing of appearance regarding the global oscillatory reference. Source: Adapted from [94].

phase coding plays a critical role in swift response for object identification and categorization tasks.

Regarding burst coding, the information is encoded considering both the number of spikes and the inter-spike interval. This allows controlling the precision in information transmission, since longer bursts (i.e., with more spikes) can carry more information. Burst spikes consist of low-duration, high-frequency spike trains that allow systems to transfer information more accurately. The presence of burst spike trains is well established in biological systems. Their function varies depending on the part of the brain we are considering. For example, in the hippocampus, these bursts play a role in memory maintenance, while in the thalamus they perform a ''wake-up call'' to prepare the neurons to receive a stimulus [95].

In summary, the mammalian brain uses different information encoding methods depending on the tasks considered. Therefore, in SNN-based architectures, the same principle applies. However, more work is needed since, on one side, the exact mechanisms by which the biological neurons encode information is not completely known. On the other side, it remains to clarify in which situations each strategy is more suitable. Rate-based coding is common in SNNs research, mainly when the focus is not the information coding itself but rather the study of a specific network module or training strategy, such as the learning rule. These methods are not the most biologically plausible, but rather straightforward to implement. But temporal-based methods are also frequently addressed due to biological plausibility and fast inference response. For example, [96] propose a modified TTFS encoding strategy (T2FSNN), having obtained an accuracy of 91.43% on the CIFAR-10 [12] dataset. On the other hand, [97] hypothesized that using burst coding in deep SNN networks is advantageous, as it increases energy efficiency and reduces latency. Their work proposes a hybrid neural coding scheme and a 2-layered network, with each layer possessing an independent neural coding strategy. The authors observed that burst coding in the hidden layer decreased the latency, regardless of the input layer neural coding. The authors report a 91.41% accuracy on the CIFAR-10 [12] dataset and 99.25% on MNIST [11]. These studies, however, took advantage of ANN to SNN conversion methods, meaning little work was done towards studying and integrating different encoding schemes with SNNs developed from scratch. But, since SNNs are still in their infancy, the most effective way to combine the encoding method with neuron model and network architecture is yet to be discovered. Although this is of great importance since if not properly encoded, information might be lost when being propagated through the network layers.

V. LEARNING STRATEGIES

SNNs aim to be a fault-tolerant, energy and data-efficient biologically inspired solution, but despite its promising results, there are some barriers to overcome. Namely, whilst it is well established that biological neurons process and transmit information in the form of spikes, the exact mechanisms by which biological neural networks learn are nonetheless an open research question. Inevitably, learning strategies in SNNs are coupled with the various elements of a neural network, including how information is encoded, the neuron model, and the general architecture. Learning in SNNs is, thus, a challenging task, and there is a need of finding an optimum solution.

Much of the success of ANNs comes from the BP algorithm [98]. In the 1990s, it was thought that learning useful representations from raw data was not feasible but later, in 2006, a team of researchers demonstrated that BP works remarkably well to train deep ANNs for classification and recognition tasks [99]–[102]. Since then, the interest in BP grew exponentially, and, today, it is undoubtedly the most widely used strategy to train DL algorithms [103]. However, there are strong arguments defending that BP is not biologically plausible. On one hand, it is thought that biological neurons perform both linear and nonlinear operations, whilst BP consists of only linear mechanisms [104] (i.e., in each iteration, a first-order Taylor approximation of the function to be optimized is applied). In second, the FB path would possess the symmetric weight of the forward propagation, which does not occur in biological systems, the so-called weight transport problem [105]. Also, BP would require bidirectional synapses whereas biological presynaptic neurons connect unidirectionally to their postsynaptic equivalents and, besides, learning in the brain occurs continuously, in a single step, whereas BP is a 2-step algorithm. Further, BP would require neurons to store the exact derivatives of the activation functions but how these derivatives could be computed and stored by neurons is currently unclear [104]. Adding to this, the activation functions of biological neuron models' (e.g., LIF) is non-differentiable, and BP is instantaneous. On the contrary, neural networks, due to the nature of spike trains, perform asynchronous computations, in a timely manner. Lastly, despite some evidence confirming the existence of some form of BP in the brain [106]–[109], the exact mechanisms by which it occurs are poorly understood [110]. For these reasons, it is challenging to train SNNs using BP, as is normally done with state-of-the-art CNNs.

A. CONVERSION OF CONVENTIONAL ANNs TO SNNs

Broadly speaking, SNNs can be divided into 2 dominant categories: Converted ANNs to SNNs and directly trained SNNs. In the first case, conventional ANNs are fully trained, using BP, before being converted to an equivalent model consisting of spiking neurons. This method is often referred to as rate-based learning since, commonly, the analogue outputs of the traditional ANNs are converted to spike trains through rate encoding. Directly trained SNNs are trained resorting to biologically plausible learning rules or use approximations to allow BP, but in either case, they consider full advantage of spiking neurons.

Naturally, converted ANNs to SNNs usually achieve performances comparable to state-of-the-art ANNs. Nonetheless, their accuracy is still behind that of non-spiking ANNs of approximate architecture [111]. Conversion could explain said performance since it assumes that the firing rate of SNNs is equal to the activation of ANNs, which is not necessarily true and thus might be a source of error. Other disadvantages of such a learning approach include: not being biologically plausible, as well as the limited implementation of many ANNs operators that are crucial to improving the performance of the networks, like max-pooling, batch normalization, or softmax activation function [112]. This means that converted ANNs to SNNs make many approximations that reduce the generalizability of conversion methods [112]. Another disadvantage resides in rate-based coding, in which computational costs increase linearly with the firing rate. Ultimately, SNNs' efficiency could be dampened, in very deep architectures or in situations where many neurons activate or possess high firing rates [112]. To this end, work has been done towards near-lossless conversion [111]–[116], nonetheless, direct training of SNNs is often preferred, so researchers can take full advantage of SNNs' properties. Concerning direct training of SNNs, ubiquitous ANNs training paradigms, like SL, reinforcement learning, and UL have been explored [17].

B. UNSUPERVISED LEARNING

Initial works propose the use of STDP, a form of UL with a local learning rule that is both biologically plausible and adequate to deal with the non-differentiable discrete binary spikes, characteristic of SNNs. As seen previously, STDP arises from Hebbian learning theories, and it states that a synaptic weight is potentiated or depressed, depending on tight correlations between pre-and postsynaptic activations. However, often, some variant of STDP is preferred to the detriment of the formal definition (Equation 4). Predominantly, this is because it is challenging to assess the precise timing of postsynaptic action potentials. On the other hand, from a biological perspective, although STDP depends on the precise timing of spikes, there exist other properties that slightly change between distinctive types of synapses, meaning there is a remarkable diversity of STDP rules [117].

One of the most frequently cited unsupervised STDP algorithms is that of [45]. The authors suggest a fully unsupervised, biologically plausible, shallow algorithm to perform a classification task on the MNIST [11] handwritten digits dataset. Inspired by the work of [118], they use a variant of STDP that resorts to synaptic traces (Equation 9) to compute weight dynamics, whilst arguing it improves simulation speed. Furthermore, they test the proposed architecture with 3 other learning rules. On one hand, they added an exponential weight dependence to the previous strategy of synaptic traces [119], [120] (see Equation 10). Subsequently, they considered pre-and postsynaptic traces, with independent weight updating for pre-and postsynaptic activity (Equation 11) and, at last, they resorted to the triplet STDP [121] with divisive weight normalization [45].

$$\Delta w = \eta \left(x_{pre} - x_{tar}\right)\left(w_{max} - w\right)^{\mu} \tag{9}$$

$$\Delta w = \eta_{post} \left(x_{pre} \mathrm{e}^{-\beta w} - x_{tar} \mathrm{e}^{-\beta (w_{max} - w)}\right) \tag{10}$$

$$\begin{cases} \Delta w = -\eta_{pre} x_{post} w^{\mu} \\ \Delta w = \eta_{post} \left(x_{pre} - x_{tar}\right)\left(w_{max} - w\right)^{\mu} \end{cases} \tag{11}$$

Notably, they added further biologically plausible mechanisms, like inhibitory synapses, to achieve competitive learning, thus promoting separability of the classes and improving the overall performance of their network. Likewise, the work of [122] introduces the lattice map spiking neural networks (LM-SNNs) where STDP is used in conjunction with the Self Organizing Map (SOM) [123] algorithm. The pointed architecture is, in many ways, similar to [45] but instead of a fixed inhibition, it introduces the notion of relaxed inhibition that increases with interneuron distance. The authors argue this encourages neighbouring neurons to learn similar filters, thus conferring the network the capacity to learn while seeing limited examples. Remarkably, the network is trained with a 2-level inhibition scheme, where after a set of predefined samples, interneuron inhibition increases suddenly, favouring competition and, consequently, augmenting model performance.

But UL based on STDP and its variants limits the models to shallow architectures with limited expressive power and that could not scale well in larger real-world problems [75], [124], [125]. Therefore, proper strategies must be found to permit unsupervised training of deep SNNs. A simple yet powerful solution is to train unsupervised deep SNNs in a layer-wise manner. For example, [126] train an unsupervised 3 layered model based on this paradigm. Interestingly, they combine a weight-dependent variant of STDP with a simplified Bayesian neuron model to classify the MNIST [11] handwritten digits dataset, yielding competitive results. Likewise, [33], [127], [128] propose a greedy layer-by-layer training scheme. To note that these works perform classification of the extremely simple MNIST [11] handwritten digits dataset, meaning that, despite promising, there is no guarantee that these models would work with more realistic data. We argue there's an unmet need of developing and validating SNN models on complex datasets like CIFAR-10 [12], CIFAR-100 [12], or ImageNet [129].

C. SUPERVISED LEARNING

SL is an appealing learning scheme and demonstrated to be efficacious for training traditional ANNs. However, due to inherent incompatibilities with spiking neurons, researchers must come up with proper strategies to apply BP to deep SNNs. To this end, much work has been done, and some strategies have proven successful. However, almost all the solutions implement approximations for the activation functions. Computations are thus performed with surrogate gradients that inevitably present approximations, causing some loss of performance. Another disadvantage of training SNNs with BP is the lack of biological plausibility. Nonetheless, BP has allowed unprecedented results of SNNs in various benchmarking datasets. The work of [130] suggests an approximate derivative algorithm that accounts for the leaky behaviour of LIF neurons. Interestingly, they leverage the central ideas behind well-known CNN architectures, like the dropout strategy, and introduce them in their model. The proposed method permits the training of deep multilayer SNNs, namely VGG and Residual architectures, and the application of spike-based BP. More specifically, the authors set the last layer neurons' threshold to a high value so that the neurons do not fire. Then, they define the output of the last layer as the accumulated voltage divided by T time steps. With this strategy, they compute a loss function, defined as the squared error over all output neurons (Equation 13), where the error (Equation 12) is computed as the difference between the target label and the output. Next, to propagate the error to hidden layers, this work approximates each output neuron activation function as equivalent to the total input current received by the neuron over T time steps (Equation 14).

$$Output_{error}, e_j = output_j - label_j \tag{12}$$

$$Loss, E = \frac{1}{2}\sum_{j=1}^{n^L} e_j^2 \tag{13}$$

$$V_{mem,j}^{L}(T) \approx \sum_{i=1}^{n^{L-1}} \left(w_{ij}x_i(T)\right) \; (output\ layer) \tag{14}$$

Regarding the hidden layers, they propose post-spike trains as neuronal outputs, with a pseudo-derivative, assuming the following approximations. In the first place, they estimate the derivative of an Integrate and Fire (IF) neuron. Next, a leak correctional term is estimated to compensate for the leaky behaviour of the LIF neurons. Finally, they obtain the approximate derivative of the LIF neuron activation function as the combinations of the 2 previous estimations. With this strategy, they can train deep multilayer SNNs and achieve competitive performances. Notably, they compare their results with ANNs to SNNs methods, demonstrating that the considered solution achieves comparable inference efficiency in terms of spikes per inference. Their results underline that deep SNNs (like VGG9 and ResNet11) are more efficient than ANN-SNN converted networks. More important, their work suggests that direct training of SNNs, resorting to spike-based BP, requires less computational energy when compared to converted ANNs to SNNs. Furthermore, it highlights the potential of SNNs for energy-efficient computations.

Another work revolves around SL based on temporal coding [131]. To overcome the limitations of discrete rate encoding schemes, the authors resort to a temporal encoding strategy, where information is encoded in the spike times, thus a continuous representation that is favourable to apply BP. Moreover, it is debated that this strategy is more efficient than rate encoding since power consumption decreases with smaller firing rates [116], [131]–[133]. They use non-leaky IF neurons with synaptic current kernels, which means that the current increases instantaneously, at the moment of arrival of an input spike, but decays exponentially afterwards. To ensure a single spike is emitted per neuron after firing, the neurons are set to an infinitely long refractory period. To define a neuron's activation function, the authors establish the set of spikes that triggered a firing as the Causal Set of Input Spikes (C). Resorting to said set, they can subsequently determine a nonlinear relationship that maps input spike times to the spike time of the firing neuron. This way, a differentiable cost function can be imposed and the gradient, with respect to the weights in the shallower layers, computed by backpropagating the errors through the network.

[134] introduced Deep Continuous Local Learning (DECOLLE), a learning approach focused on local error functions. To compute the local errors, the authors use intermediate classifiers with fixed random weights and auxiliary random targets, $\hat{y}$. The inputs to these classifiers consequently represent the activations of the layer being trained. Moreover, instead of minimizing a global objective function, the algorithm minimizes many local objectives, yet, this approach still allows the network to minimize the loss at the top layer. In addition, it puts pressure on deeper layers to learn relevant representations whilst leading the network to learn a stack of useful hierarchical features [131]. To enforce locality, DECOLLE sets to zero all non-local gradients. Errors are propagated to only update the weights of the connections incoming to the local spiking layer, but the overall approach can be interpreted as a synthetic gradient without an outer loop to mimic a full BP. They trained a fully connected SNN with three convolutional layers and Poisson encoding, reporting an error of 4.46 % on the DVS128-Gesture [135] dataset.

In turn, [136] developed a SL strategy based on TTFS coding. This approach allows the authors to derive an analytical expression for the time, T, at which the membrane voltage will first cross the threshold. The expression, T, is thus differentiable with respect to synaptic weights and presynaptic spike times, meaning BP can be used. Plus, it allows the exact computation of partial derivatives, and the fact that each neuron only spikes once is extremely desirable from an efficiency standpoint. Another work that takes advantage of temporal coding for computing the exact derivatives of postsynaptic with respect to presynaptic times for BP is that of [137]. The adopted neuron model was the Spike Response Model (SRM), with an alpha function [138] to model the synaptic conductance and consequently the membrane potential of the postsynaptic neuron. This allows computing the spike time of an output neuron with respect to the presynaptic spike times. The desired behaviour to predict a class is that the neuron

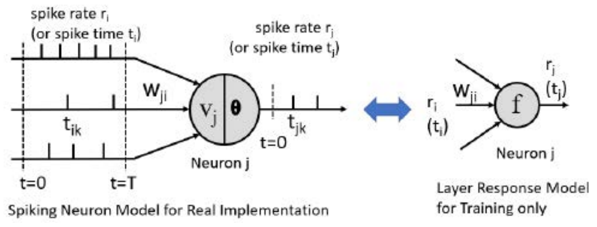


FIGURE 11. **Difference between the neuron model for neuromorphic hardware implementation and the layer response neuron model for training. Source: [139].**

of the correct class should be the first to spike. To get the prediction error, the softmax function is computed on the negative values of the output spike times, which minimises the spike time of the target neuron while maximising the spike time of the non-target neurons. Next, the cross-entropy loss is calculated in the customary form.

A major drawback when applying BP to SNNs is the need of computing the membrane voltage for each neuron, at each time step, which is computationally prohibitive in current hardware systems, despite it being efficient when implementing SNNs in neuromorphic hardware. Additionally, there is the problem of the non-diferentiable LIF neurons' activation function. To circumvent that need, the work of [139] suggests an abstract layer response model of the neurons for training deep networks, as illustrated in Figure 11.

Considering a single layer, l, the algorithm starts by defining the layer response model input and output as, respectively,

$$z_{l-1,i} = e^{t_{l-1,i}/\tau}$$

and

$$z_{l,j} = e^{t_{l,j}/\tau},$$

with $t_{l,j}$ the time to a neuron's first and only spike as defined by TTFS coding, and i/j the input/output neurons. Then, the output of the L layered network, Z_L is defined by a non-linear mapping, f, and connection weights, W, that establish $Z_L = f(Z_0, W)$. This allows to obtain the loss, $\mathcal{L}$ for target class, C, as:

$$\mathcal{L}(z_L, C) = -log\frac{z_{L,C}^{-1}}{\sum_{i\neq c} z_{L,i}^{-1}} + K\sum_{l=1}^{L}\sum max\{0, \theta\} - \sum_i w_{ji}^l + \lambda\sum_{l=1}^{L}\sum_{j,i}(w_{ji}^l) \quad (15)$$

Finally, considering that the layer response model uses the closed-form input-output response $t_j = f(t_i; w_{ji})$, it becomes possible to apply BP as is traditionally done in ANNs. The learned weights, w_{ij} can then be transferred to IF neurons for inference and hardware implementation. This algorithm extends on the work of [131], but overcomes some of its limitations. Namely, whereas [131] was limited to shallow networks (2-3 fully connected layers), [139] propose a new training scheme that improved computation speed and convergence, allowing deep SNNs to scale.

The work of [140], on the other hand, explores latency learning to avoid the derivation of the thresholding activation function. It considers TTFS coding, with one spike per neuron, and IF neurons. At the output layer, there is only 1 neuron per class, where the predicted class is that of the first spiking neuron. This is a very sparse and energy-efficient coding scheme. The network is trained with a temporal form of BP, with surrogate gradients for the neuron firing time with respect to its membrane potential, where the error is computed considering the difference between the firing time and target firing times. Interestingly, target firing times are defined dynamically, they propose a relative method that takes the actual firing times into account. Assuming an input image of the i^{th} category, the minimum output firing time, τ, is computed as $\tau = min\{t_j^o | 1 < j < C\}$. Then, the target firing time is set as:

$$T_j^o = \begin{cases} \tau & \text{if } j = i, \\ \tau + \gamma & \text{if } j \neq i \;\&\; t_j^o < \tau + \gamma, \\ t_j^o & \text{if } j \neq i \;\&\; t_j^o \geq \tau + \gamma \end{cases} \quad (16)$$

where γ is a positive constant term penalizing output neurons with firing time close to τ. In a special case where neurons are silent during the entire simulation time, the target is defined as:

$$T_j^o = \begin{cases} t_{max} - \gamma & \text{if } j = i, \\ t_{max} & \text{if } j \neq i \end{cases} \quad (17)$$

to promote the firing of the output neuron during the simulation. Although a simple solution, the proposed strategy was demonstrated to achieve competitive performance on the MNIST [11] and CALTECH face/motorbike datasets. Given the very sparse nature of this algorithm, the authors suggest it could be particularly energy and memory efficient, specially when combined with neuromorphic hardware. Also, it can make accurate and quick predictions, as a decision can be made before all neurons have fired, way earlier than the entire stimulus presentation time, contrary to rate-based SNN models.

Recently, [141] introduced EventProp, a novel, and promising event-based method, that employs BP, but that allows the computation of exact gradients. In essence, the authors backpropagate errors at spike times to obtain the exact gradient in an event-based, temporally, and spatially sparse manner. More specifically, to compute the gradient the algorithm starts by considering the LIF neuron model as a dynamic system, where the partial derivative of a state variable ($V(t)$) with respect to a parameter, p (synaptic weight, w), jumps at discontinuities. Next, the adjoint method [142] is combined with the partial derivative jumps of the LIF neuron to derive the EventProp algorithm, which the authors define as analogous to BP in ANNs. Since EventProp backpropagates errors at spike times, it only requires the storage of variables at spike times. Therefore, it has low memory requirements, thus favourable for neuromorphic computing. In fact, contrary to the previous strategies that require the computation of gradients for every neuron at each time step, EventProp only requires computations for spiking neurons, thus possibly being more energy-efficient than other works.

In contrast with biological neurons that present dynamic membrane properties(e.g., heterogeneous time constants, adaptive thresholds), most existing learning algorithms require manual tuning of membrane-related parameters and assume that all neuron populations in a SNN present the same values for those constants. Nonetheless, to achieve a more biologically plausible algorithm, [143] introduced the Parametric Leaky Integrate-and-Fire (PLIF) neurons, which present learnable time constants, τ. This work's training framework is supported on a BP strategy. Defining the neurons' output as O, and the target as Y, the loss function was defined as the Mean Squared Error (MSE(O, Y)), considering that the neuron that represents a given class should have the maximum excitability, whereas the others should remain silent. Then, the authors compute the gradients resorting to a surrogate activation function, defined as $\frac{1}{\pi}\arctan(\pi x) + \frac{1}{2}$. The results were evaluated on neuromorphic datasets, like CIFAR10-DVS [144] or DVS128-Gesture [135], and suggest PLIF-based SNNs to learn faster and to achieve better performances when compared to LIF-based SNNs.

In the same line of reasoning, [145] introduced an SNN with direct input encoding and leakage and threshold optimization (DIET-SNN). The proposed learning scheme resorts to a hybrid strategy where an ANN is first converted to SNN before being fine-tuned with surrogate gradient and BPTT. Interestingly, weighted pixel values are directly fed to the network's first layer, at each time step, instead of being converted to spikes. The first layer of the network thus works as both feature extractor and spike generator. Moreover, the underlying training strategy, supported on surrogate gradients and BPTT, optimizes not only the network parameters (connection weights), but also the LIF neuron parameters, namely, firing threshold and membrane leak factor. The authors suggest the neuron threshold to be an important parameter as if too high, it prevents the neuron from firing (dead neuron problem) and if too low, it affects the ability of the neuron to distinguish between input patterns. On the contrary, the optimized membrane leak makes the network firing response less sensitive to irrelevant input and increases the sparseness of convolutional and dense layers. DIET-SNN was tested on the CIFAR-10 [12], CIFAR-100 [12], and ImageNet [129] datasets and demonstrated to achieve better latency/accuracy tradeoff with $20-500$x less timesteps.

Importantly, BP has also permitted the training of very deep ANNs. In conventional ANNs, deep residual networks, which consist of many stacked "Residual Units", have achieved top performance. Identity mapping is a central idea in residual learning, where the output of layer (l), (X_{l+1}), is given by $(F(X_l, W) + X_l)$, with X_l the input feature vector, and $F(X_l, W)$, the residual mapping to be learned. The $\bigoplus$ operator is realized by skip connections that simply perform identity mapping. With this strategy, ResNet [3] was presented as a robust and reliable solution to the

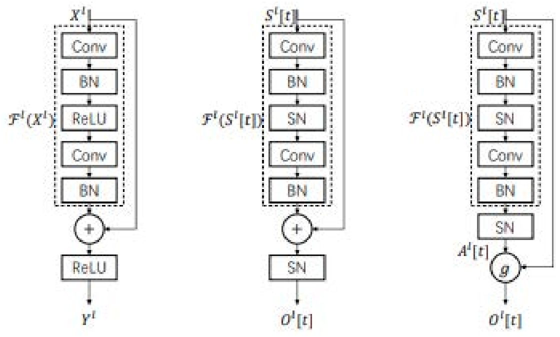


FIGURE 12. Residual modules of, from left to right: ResNet, Spiking ResNet, and SEW ResNet. Source: [148].

vanishing/exploding gradient problem in deep networks. Similarly, converted ResNets to Spiking ResNets have achieved competitive performance [146], [147], although requiring longer time steps to achieve top results, as conversion is based on rate coding. To overcome the limitations of conversion strategies, [148] introduced Spike-Element-Wise ResNet (SEW ResNet), a direct training strategy to allow residual learning in SNNs. Much like Spiking ResNet [146], SEW ResNet substitutes the ReLU activation for a Spiking Neuron (SN), however, it also finds an element-wise function, g, to realize identity mapping. This strategy overcomes the drawbacks of Spiking ResNet, namely the exploding/vanishing gradient problem and the limited applicability of Spiking ResNet to specific neuron models/dynamics. Figure 12 illustrates the main differences between a ResNet, Spiking ResNet, and SEW ResNet module. Combined with spike-based BP this strategy has allowed, for the first time, the training of very deep SNNs (with more than 100 layers) and achieved competitive results.

The focus of this work is not that of reinforcement learning problems, but the strategy has equally been used to train SNNs [149], [150]. It comes from biological evidence supporting that neuromodulators impact learning in the brain. For example, it was demonstrated that dopamine acts as a reward signal, affecting synaptic plasticity [151]–[153].

In summary, several strategies have been adopted to train SNNs, with some more successful than others. For a complementary and comprehensive overview of learning in SNNs we refer the reader to the work of [17]. However, we have seen that learning in SNNs is an open research question that presents many challenges mainly due to the non-differentiable nature of SNN activation functions, sparsity of spike trains, and computations over time. Many of the proposed solutions resort to approximations, yet this translates to limited generalizability, meaning more effort is needed from the research community towards a unified strategy for training SNNs.

VI. RESULTS AND DISCUSSION

Many SNNs algorithms demonstrate good results on simple datasets like the MNIST [11] handwritten digits dataset, yet,

TABLE 1. Summary of reviewed algorithms.

Architecture	Reference	Learning	Learning Method	Encoding	T	Dataset	Accuracy
CCVDN1	[112]	Sup.	Converted CNN	Rate	-	MNIST	99.40%
CCVDN1	[112]	Sup.	Converted CNN	Rate	-	CIFAR-10	90.80%
CCVDN1	[112]	Sup.	Converted CNN	Rate	550	ImageNet	70.60%
Conv. ResNet-20	[154]	Sup.	Converted CNN	Rate	400-600	CIFAR-10	93.58%
Conv. VGG-16	[154]	Sup.	Converted CNN	Rate	400-600	CIFAR-100	70.55%
Conv. VGG-16	[154]	Sup.	Converted CNN	Rate	512	ImageNet	72.34%
Spiking ConvNet	[111]	Sup.	Converted CNN	Rate	500	MNIST	99.10%
HSNN	[71]	Unsup.	STDP	Rate	300	MNIST	96.30%
Ensemble HSNN	[33]	Unsup.	Variable Threshold + STDP	Rate	-	MNIST	99.27%
Ensemble HSNN	[33]	Unsup.	Variable Threshold + STDP	Rate	-	CIFAR-10	93.00%
Deep CSNN	[76]	Unsup.	Variable Threshold + STDP	Temporal	30	Caltech	99.10%
Deep CSNN	[76]	Unsup.	Variable Threshold + STDP	Temporal	30	ETH-80	82.80%
Deep CSNN	[76]	Unsup.	Variable Threshold + STDP	Temporal	30	MNIST	98.40%
SNN+Localized RFs	[77]	Unsup.	STDP	Rate	200	MNIST	98.90%
SNN+Localized RFs	[77]	Unsup.	STDP	Rate	200	CIFAR-10	55.60%
GLSNN	[47]	Sup.	STDP+Global FB	Temporal	10	MNIST	98.62%
GLSNN	[47]	Sup.	STDP+Global FB	Temporal	10	F-MNIST	89.05%
FB SNN	[48]	Sup.	Implicit Differentiation	Rate	30	MNIST	99.59%
FB SNN	[48]	Sup.	Implicit Differentiation	Rate	5	F-MNIST	90.25%
FB SNN	[48]	Sup.	Implicit Differentiation	-	30	N-MNIST	99.47%
FB SNN	[48]	Sup.	Implicit Differentiation	Rate	100	CIFAR-10	92.82%
FB SNN	[48]	Sup.	Implicit Differentiation	Rate	100	CIFAR-100	73.43%
LM-SNN	[122]	Unsup.	STDP-SOM	Rate	350	MNIST	94.07%
SNN w/ Stable STDP	[116]	Unsup.	Stable STDP	Rate	25	MNIST	85.00%
Deep SNN+Adapt. Thresh.	[127]	Unsup.	STDP	Rate	100	MNIST	96.60%
Multilayer SCNN	[128]	Unsup.	STDP	Rate	20	MNIST	96.97%
SNN with spike-based BP	[130]	Sup.	BP w/ approx. derivative	Rate	50	MNIST	99.59%
SNN with spike-based BP	[130]	Sup.	BP w/ approx. derivative	-	100	N-MNIST	90.09%
SNN with spike-based BP	[130]	Sup.	BP w/ approx. derivative	Rate	100	CIFAR-10	90.95%
Sup. SNN+Temp. Coding	[131]	Sup.	BP with causal input spikes	Temporal	-	MNIST	97.14%
EventProp	[141]	Sup.	Adjoint+partial deriv. jumps	Temporal	-	MNIST	97.61%
BP-STDP	[155]	Sup.	STDP-based BP	Rate	-	MNIST	97.20%
BP-STDP	[155]	Sup.	STDP-based BP	Rate	-	IRIS	96.00%
PLIF-neuron SNN	[143]	Sup.	Spike-based BP	-	8	MNIST	99.72%
PLIF-neuron SNN	[143]	Sup.	Spike-based BP	-	8	F-MNIST	94.38%
PLIF-neuron SNN	[143]	Sup.	Spike-based BP	-	8	CIFAR-10	93.50%
PLIF-neuron SNN	[143]	Sup.	Spike-based BP	-	10	N-MNIST	99.61%
PLIF-neuron SNN	[143]	Sup.	Spike-based BP	-	20	CIFAR10-DVS	74.80%
PLIF-neuron SNN	[143]	Sup.	Spike-based BP	-	20	DVS128-Gest	97.57%
DIET-SNN	[145]	Sup.	Hybrid	-	5	CIFAR-10	92.70%
DIET-SNN	[145]	Sup.	Hybrid	-	5	CIFAR-100	69.67%
DIET-SNN	[145]	Sup.	Hybrid	-	5	ImageNet	69.00%
SEW-ResNet-34	[148]	Sup.	Spike-based BP	-	4	ImageNet	67.04%
SEW-ResNet-50	[148]	Sup.	Spike-based BP	-	4	ImageNet	67.78%
SEW-ResNet-101	[148]	Sup.	Spike-based BP	-	4	ImageNet	68.76%
SEW-ResNet-152	[148]	Sup.	Spike-based BP	-	4	ImageNet	69.26%
SEW-ResNet-ADD	[148]	Sup.	Spike-based BP	-	16	DVS128-Gest	97.92%
SEW-ResNet	[148]	Sup.	Spike-based BP	-	16	CIFAR-10-DVS	74.40%
Temporal-Coded Deep SNN	[139]	Sup.	Direct Training w/ LRM	Temporal	-	MNIST	99.33%
Temporal-Coded Deep SNN	[139]	Sup.	Direct Training w/ LRM	Temporal	-	CIFAR-10	92.68%
Temporal-Coded Deep SNN	[139]	Sup.	Direct Training w/ LRM	Temporal	-	ImageNet	68.80%
S4NN	[140]	Sup.	Temporal BP	Temporal	256	Caltech	99.20%
S4NN	[140]	Sup.	Temporal BP	Temporal	256	MNIST	97.40%

most papers do not address more complex data. On the other hand, SNNs require extensive fine-tuning of hyperparameters, and this task can have a significant impact on learning and subsequent models' accuracy. In addition, most papers do not report the set of used hyperparameters, which hinders the reproducibility of the methodology. Overall, there is a need of understanding general practical considerations when implementing SNN models. We argue this knowledge

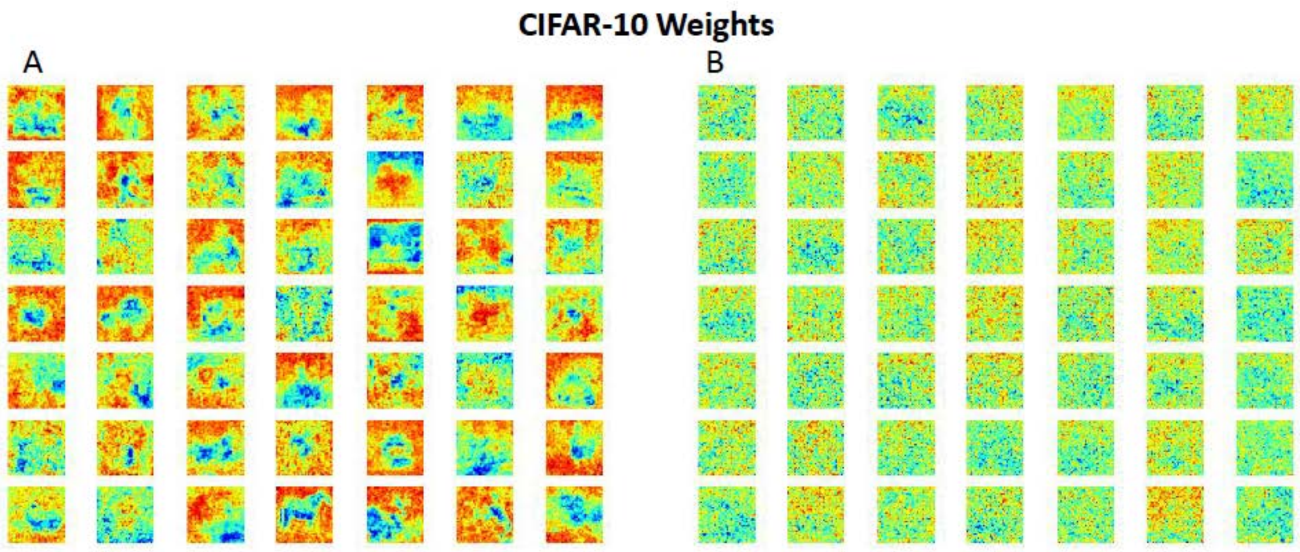


FIGURE 13. A: Neurons' RFs (selected randomly and rearranged to the input images shape) of the best performing model on the training set. The RFs are well defined. It is even possible to identify some classes (e.g. horse; car) B: The same neurons' RFs after 1800 training iterations. We observe that after several iterations the learned RFs are simple and non-interpretable. This could be explained by the high degree of variability in the CIFAR-10 dataset, where averaging effects lead to neurons that respond to a wide range of prototypes.

could help researchers achieve better and faster results when developing custom architectures. Henceforth, we here implement 2 learning algorithms: SL with surrogate gradients and unsupervised STDP. For STDP, we replicate a widely cited algorithm [45].In respect to SL, we resort to 3 network architectures with varying degrees of depth and complexity. The goal is to provide the reader with some insights about practical considerations when implementing SNNs and to compare supervised and unsupervised training strategies. We further discuss the advantages and disadvantages of each strategy while exploring the relevance of hyperparameter optimization and model performance on different computer vision datasets. In this case, we opted for the MNIST [11] and CIFAR-10 [12] datasets.

Table 1 demonstrates some details and the reported performance of the previously reviewed algorithms. We observe that most works use the MNIST [11] handwritten digits dataset. At the same time, performance usually decreases in more complex data. Notably, [33] report a performance of 93.00% on the CIFAR-10 [12] dataset resorting to an UL strategy. In general, SL and converted ANNs to SNNs demonstrate superior performance, but lack of biological plausibility and limitations in the ability to generalize make these strategies less attractive.

For replicating the work of [45], we opted for the BindsNET [156] package. Moreover, we introduced the dropout [157] strategy to assess how this regularization technique would impact learning. The choice of SNN simulation software is critical when developing SNN models. Several tools exist, but most of them are not ML oriented or present steep learning curves for new users [156]. On the other hand, many are biology-oriented, meaning they are computationally expensive to simulate and need extensive hyperparameter tuning [156]. Thus, [156] have developed BindsNET, an ML oriented library in *Python*, built on top of PyTorch [158]. BindsNET contains a set of software objects and methods to simulate diverse types of neurons (bindsnet.network.nodes), as well as various types of connections between them (bindsnet.network.topology). The bindsnet.network.Network object is responsible for combining the different nodes and connections, whilst also coordinating the simulation logic of all underlying components [156]. We opted for this package since it allows rapid prototyping and permits GPU acceleration. On the other hand, BindsNET

TABLE 2. The impact of hyperparameters and the dropout strategy on the performance of the SNN architecture proposed by [45].

Dataset	Exc. Strength	Inh. Strength	N. Neurons	Prob. of Drop.	Accuracy
MNIST	22.50	17.50	100	-	51.80%
MNIST	22.50	17.50	400	-	67.74%
MNIST	22.50	17.50	800	-	75.03%
MNIST	22.50	120.00	100	-	72.00%
MNIST	22.50	120.00	400	-	90.28%
MNIST	22.50	120.00	800	-	91.41%
MNIST	64.00	32.00	100	-	61.76%
MNIST	64.00	32.00	400	-	76.80%
MNIST	64.00	32.00	800	-	81.43%
MNIST	22.50	120.00	400	0.10	90.63%
MNIST	22.50	120.00	400	0.20	90.73%
MNIST	22.50	120.00	800	0.10	90.91%
MNIST	22.50	120.00	800	0.20	91.31%
MNIST	64.00	32.00	400	0.10	76.60%
MNIST	64.00	32.00	400	0.20	80.20%
MNIST	64.00	32.00	800	0.10	83.00%
MNIST	64.00	32.00	800	0.20	83.80%
CIFAR-10	22.50	120.00	400	-	12.89%
CIFAR-10	64.00	32.00	400	-	10.00%
CIFAR-10	22.50	17.50	400	-	10.81%

requires less hyperparameter fine-tuning in comparison with the Brian [159] simulator.

For dropout regularization, instead of randomly dropping connections, we randomly forced input pixel intensities to zero, during training, with a probability given by p_{drop}. The subsequent postsynaptic neurons would not activate, and consequently, the connection weight would not be updated. This ensures the same units are dropped during the entire duration of the example presentation and avoids averaging effects [130]. Table 2 presents the obtained performances. It allows the comparison between models trained with different hyperparameters and different datasets. Furthermore, we observe the effects of dropout regularization in the training of SNNs.

Our results are consistent with the performances reported by [45]. We observe the models with more neurons have higher accuracy. Besides, introducing the dropout strategy also produces significant performance increases. A challenge with SNNs, and with this model, in particular, is the choice of hyperparameters. To assess the impact of excitatory to inhibitory and inhibitory to excitatory synaptic strengths in model performance, we employed a random search algorithm to identify the most proper set of values for excitatory and inhibitory strengths. Interestingly, the obtained performances are significantly lower than using the values provided by the BindsNET authors in their examples directory (excitatory strength=22.50 and inhibitory strength=120), but significantly better than the default parameters of the BindsNET implementation of the [45] algorithm (excitatory strength=22.50 and inhibitory strength=17.50). The suboptimal parameters detected by our algorithm could be explained by the nature of random search, that despite being more computationally efficient might not find the optimum set of hyperparameters. Broadly speaking, random search [160] translates to results as good or better than other strategies, like manual search or grid search, but given the nature of the addressed SNN architecture, these two hyperparameters impact competition and neurons' activation, meaning it directly influences learning, thus being of great importance in training the model. This underlines the challenges of hyperparameter optimization in SNNs. Plus, we observe that the algorithms trained with dropout have, in general, better performances than their no dropout counterparts, specifically considering the model architectures with a smaller number of neurons. We argue that, in this situation, the dropout regularization prevents a limited subset of neurons from dominating the others (due to competition), thus reducing overfitting. Interestingly, due to the sparse nature of SNNs, a much lower p_{drop} (0.2) is required when compared to conventional ANNs (usually around 0.5) [130]. Finally, we observe a reduced performance ($10\% \leq accuracy < 12.89\%$) when the model was trained on the CIFAR-10 [12] dataset. These results underline the challenges of training SNN models in complex datasets. Figure 13, indicates two different sets of weights learned by the network trained on the CIFAR-10 [12] dataset. In Figure 13-A, we observe a set of neurons' RFs (weights selected randomly and rearranged to the shape of the input image) of the model with the best training performance (accuracy = 18.80%), and in Figure 13-B, the same neurons' RFs after 1800 training iterations (accuracy = 9.74 %). It clearly demonstrates that in datasets with a considerable degree of variability, the averaging effect of this SNN architecture leads to simple and non-interpretable receptive fields, where a single neuron can respond to a wide range of prototypes. This results in high confusion between classes and poor model performance. To further illustrate the challenge of training complex data using shallow SNN models, in Figure 14 we show there is an overlap of learned RFs between 2 distinct classes. The weights corresponding to high-intensity regions (higher firing rates) change the most when compared to smaller firing rate input stimuli. In fact, the most impressive performances reported in the literature of models trained on complex datasets usually resort to SL strategies like approximate BP or converted ANNs to SNNs, or use some form of CNNs, meaning there's an unmet need of finding the best strategies to train UL SNN models that can compete with traditional ANNs.

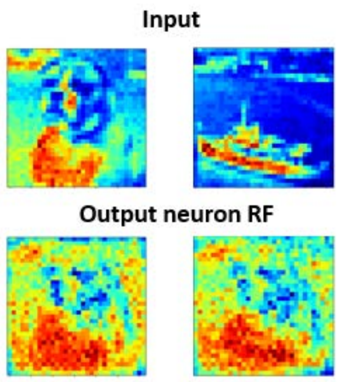


FIGURE 14. **Top: Input images of two distinct classes (dog; boat). Bottom: corresponding learned RFs. It becomes clear that high intensity image regions dominate the activation of the excitatory layer neurons, meaning that if there is a high degree of overlap between a previously learned RF and a high intensity region of a new image, that neuron has a high probability of firing again, regardless of the input class. We observe that the synaptic weights corresponding to high intensity regions (higher firing rates) change the most when compared to smaller firing rate input stimuli.**

For the SL experiments, due to ease of use and similarity with typical PyTorch workflows, we opted for the snnTorch [161] *Python* package as it is particularly designed for performing gradient-based learning with SNNs. snnTorch can be intuitively used with PyTorch [158] and is agnostic to typical neural network layers, such as fully-connected layers, convolutional layers, or residual connections. Interestingly, spiking neurons are designed in such a way that they can be easily stacked on top of other neural network layers as if they were yet another activation function. Furthermore, membrane potentials are computed recursively, meaning the gradient can be computed without storing membrane potential traces for all neurons in a system. Naturally, snnTorch is suitable for GPU-enabled training, which increases the speed of computations.

Considering specifically the MNIST [11] handwritten digits dataset, we opted for a 2-layered network, similar to [45]. We considered a fully connected hidden layer, with N neurons, for feature extraction, and an output classification layer with 10 neurons (1 per class). Then, we converted the static images to spike trains, resorting to rate coding, where, at each time step, the probability of an input spike was proportional to the pixel intensity. Each batch of sample images was presented to the network for 25 time steps. To allow BP, we considered Cross Entropy (CE) and Mean Squared Error (MSE) as loss functions, $\mathcal{L}$, respectively established by equations 18 and 19:

$$\mathcal{L}_{CE} = -\frac{1}{N}\sum_{n=1}^{N} log\left(\frac{e^{\sum_{t=0}^{T} S_{n_y}(t)}}{\sum_{j=0}^{C-1} e^{\sum_{t=0}^{T} S_{n_j}(t)}}\right) \tag{18}$$

$$\mathcal{L}_{MSE} = \frac{1}{T}\frac{1}{N}\sum_{n=1}^{N}\sum_{j=0}^{C-1}\left(\sum_{t=0}^{T} S_{nj}(t) - \sum_{t=0}^{T}\hat{S}_{nj}(t)\right)^2 \tag{19}$$

With N the batch size; C the number of classes; T the batch presentation time; $S_k(t)$ the predicted spike activity of the LIF neurons (in $\mathcal{L}_{CE}$, S_y represents the predicted spiking activity of the correct class) and $\hat{S}(t)$ the target spiking activity. To calculate the losses, the spikes are first accumulated over T time steps. The CE Count Loss, $\mathcal{L}_{CE}$, promotes the neuron of the correct class to spike at each time step, t, while suppressing the activation of the incorrect classes neurons. In turn, the MSE Count Loss, $\mathcal{L}_{MSE}$, encourages a spiking activity target for the correct class (correct rate), and a target for the incorrect classes (incorrect rate). But to make BP possible we approximate a neuron's activation function as a Sigmoid, similar to [162], which then allows to compute surrogate gradients:

$$S \approx \frac{1}{1+e^{-kU}} \tag{20}$$

$$\frac{\partial S}{\partial U} = \frac{ke^{-kU}}{(1+e^{-kU})^2} \tag{21}$$

Considering this training strategy, we assessed the impact of different hyperparameters on network performance, namely, the number of hidden neurons, the learning rate, η, and the decay rate of the membrane potential, β. Table 3 presents in detail the obtained performances for different values of the considered hyperparameters on the MNIST [142] handwritten digits dataset. Overall, we observe better performance with SL than with UL. Generally speaking, our results indicate $\mathcal{L}_{MSE}$ to be more suitable than $\mathcal{L}_{CE}$, while the decay rate, β, is also a very important hyperparameter. We obtained a top performance of 98.53 % for 6400 hidden neurons, trained with $\mathcal{L}_{MSE}$ (correct rate = 0.8, incorrect rate = 0.1), of learning rate $\eta = 5e^{-4}$ and decay rate $\beta = 0.75$. In turn, comparing the performance of the 800 hidden neuron SL network with the best results of our experiments on the Diehl and Cook SNN [45], we observe an improvement of 6.12 %.

We also ran the 2-layered shallow network on the CIFAR-10 [12] dataset. The results are very consistent with the performance of the Diehl and Cook [45] SNN on these data, suggesting deeper networks would be necessary to achieve competitive results. Therefore, we adopted 2 additional SNN architectures. The topology of the first network (CSNN), consisted of 2 5×5 convolutional layers, for feature extraction, with, respectively, 24 and 128 output channels, interleaved by a 2×2 max pooling layer, followed by a LIF neuron. On top of these 2 convolutional blocks, we added a fully connected layer for input classification. Figure 15 illustrates the considered network architecture. We then trained various models with different hyperparameters to assess their impact on model performance. We considered both $\mathcal{L}_{MSE}$ and $\mathcal{L}_{CE}$ as loss functions and approximate the neuron's activation as a fast sigmoid [162] to compute the surrogate gradients:

$$S \approx \frac{U}{1+k|U|} \tag{22}$$

As an alternative approach, in line with the work of [130], we considered a VGG9 SNN architecture (VGG9-C), composed of 8 convolutional layers and a fully connected layer for classification. We used 2×2 average pooling layers to reduce the size of convolutional feature maps, as suggested in [130], but where we considered a threshold of 0.5 for the LIF layer. As a variant of this architecture, we substitute the last convolution with a fully connected layer (VGG9-F). Figure 16 illustrates both of these strategies. Regarding the bipolar transform, we considered the 3 RGB channels. Then, we follow the approach of [130] and scale pixel intensities to the $[-1, 1]$ range so that each channel presents mean = 0 and standard deviation = 1. This ensures we can take maximum advantage of the images' informative content. At last, we separate the positive and negative pixel intensities into bipolar channels before rate encoding. For data augmentation, we considered random horizontal flips and random resized crops. We trained each model for 50 epochs, with a batch size of 128, and Adam optimizer. Additionally, we considered a correct rate of 0.8 and incorrect rate of 0.1 for $\mathcal{L}_{MSE}$. In general, we opted for a fixed threshold of 0.75 for the LIF layers but as a final experiment, we trained the VGG9-F architecture with learnable threshold and decay rate β (VGGG9-F†). In table 4 we summarize the hyperparameters and corresponding performances.

The results underline that SNNs trained with surrogate gradients present by far the best performance when compared with UL. The reported performances are inferior to the literature top results, nonetheless, we mainly intended to underline the advantages of BP with surrogate gradients, so we did not perform any hyperparameter optimization. Moreover, we apply the backward pass only once for each simulation and did not experiment with other loss functions and surrogate gradients, meaning further accuracy improvements could be achieved. We also could have considered dropout regularization as the technique is commonly used

TABLE 3. Performance of SL SNNs on the MNIST [11] handwritten digits dataset. To allow direct comparison with [45], a similar 2-layered network was used. Besides the hidden layer neurons for feature extraction, we added a classification layer composed of 10 neurons (1 per class).

N. Neurons	Loss	Corr. Rate	Incorr. Rate	Learn. Rate (η)	Decay rate (β)	Accuracy
0100	$\mathcal{L}_{CE}$	-	-	$5e^{-4}$	0.10	82.16 %
0400	$\mathcal{L}_{CE}$	-	-	$5e^{-4}$	0.10	88.53 %
0800	$\mathcal{L}_{CE}$	-	-	$5e^{-4}$	0.10	90.21 %
1000	$\mathcal{L}_{CE}$	-	-	$5e^{-4}$	0.10	89.44 %
0800	$\mathcal{L}_{CE}$	-	-	$5e^{-4}$	0.65	94.22 %
0800	$\mathcal{L}_{CE}$	-	-	$5e^{-4}$	0.75	94.88 %
0800	$\mathcal{L}_{CE}$	-	-	$5e^{-4}$	0.90	93.38 %
0800	$\mathcal{L}_{CE}$	-	-	$5e^{-4}$	0.95	85.97 %
0800	$\mathcal{L}_{CE}$	-	-	$1e^{-4}$	0.75	93.14 %
0800	$\mathcal{L}_{CE}$	-	-	$2e^{-4}$	0.75	91.02 %
0800	$\mathcal{L}_{CE}$	-	-	$2e^{-5}$	0.75	91.37 %
3200	$\mathcal{L}_{CE}$	-	-	$5e^{-4}$	0.75	95.97 %
6400	$\mathcal{L}_{CE}$	-	-	$5e^{-4}$	0.75	96.53 %
6400	$\mathcal{L}_{MSE}$	0.80	0.10	$5e^{-4}$	0.75	97.23 %
6400	$\mathcal{L}_{MSE}$	0.70	0.30	$5e^{-4}$	0.75	97.10 %
6400	$\mathcal{L}_{MSE}$	0.80	0.10	$5e^{-4}$	0.75	98.53 %
0800	$\mathcal{L}_{MSE}$	0.80	0.10	$5e^{-4}$	0.75	97.53 %
0800	$\mathcal{L}_{MSE}$	0.80	0.20	$5e^{-4}$	0.75	95.87 %
0800	$\mathcal{L}_{MSE}$	0.90	0.10	$5e^{-4}$	0.75	96.08 %

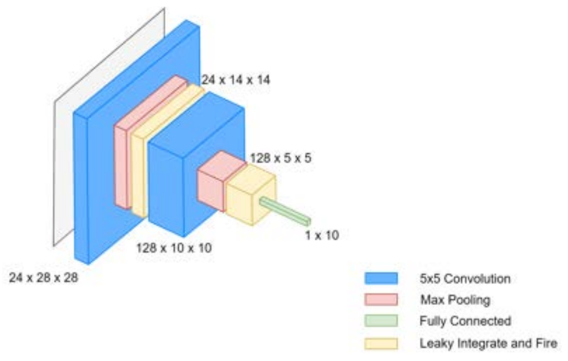


FIGURE 15. Convolutional SNN (CSNN) for classification on the CIFAR-10 [12] dataset. It consists of 2 blocks of convolutional and pooling layers interleaved by LIF neurons, for feature extraction, and a fully connected layer for classification.

in SNNs trained with surrogate gradients. However, the reported results are fairly illustrative of the advantages and disadvantages of using BP with surrogate gradients to train SNNs. We experienced a significant performance increase when the RGB input was directly encoded to spike trains instead of first converting the images to grayscale. We also demonstrate that data augmentation can be used in conjunction with SNNs trained with surrogate gradients and that it leads to a significant improvement of the results. Moreover, we demonstrate the importance of selecting the number of time steps. If too low, the outputs of each neuron will be close to each other and the network will not be able to discern between inputs, leading to poor performance. On the contrary, we also experience a drop in performance when we increase the number of time steps. Ultimately, a trade-off must be achieved between the desired performance and efficiency. In our experiments, we observe the optimum accuracy with 100 time steps. Although with 75 time steps, the models' have not achieved a significantly inferior performance, suggesting this could be the adequate number of time steps to achieve the optimum efficiency/accuracy trade-off.

In summary, our results highlight the performance gap between UL and SL in SNNs. We observe that whereas unsupervised local learning with STDP works quite well on simple datasets, the strategy is not currently feasible on complex data and more work is necessary towards achieving high performing and biologically plausible UL algorithms. On the contrary, SL with surrogate gradients offers a more optimum efficiency/accuracy trade-off. Although, for the reasons already discussed, BP is not biologically plausible, the solution can scale to more complex data. Moreover, SL requires less simulation time steps (25-125 in our experiments) to achieve competitive performance, whereas for STDP usually hundreds of time steps are required (250 in this work). Considering the 2 *Python* packages used for this work, we observe that SL with snnTorch is suitable for fast and efficient deep network training. BindsNET, in turn, is clock-driven which means the simulation takes longer as we increase the number of time steps. Not only that, but simulation time also increases with network size (number of layers, neurons, connections, etc.) Nonetheless, although both packages are ML oriented, each answers a different need. Whereas, much like the Brian simulator, BindsNet defines an abstract representation of biological neural networks with the notion of neurons and synapses as a set of objects (''node'' and ''connection'') and is adequate for unsupervised STDP and reinforcement learning, snnTorch is more suitable for SL tasks and its implementation of SNNs more closely mimics common PyTorch [158] workflows, where, typically, activation functions are replaced by biological neuron models (e.g., LIF)

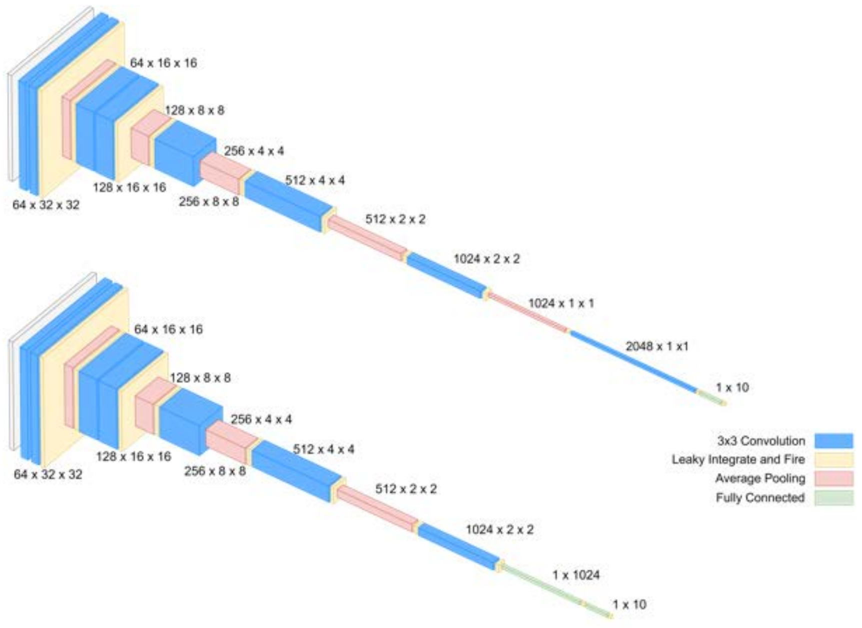


FIGURE 16. VGG9 SNNs (VGG9-C/F) for classification on the CIFAR-10 [12] dataset. The main difference between the top and bottom neural network architectures is that we replace the last convolution with a fully connected layer.

TABLE 4. Performance of SL SNNs on the CIFAR-10 [12] dataset. We observe better performances with RGB inputs and Bipolar transform. Data augmentation consisted of random horizontal flips and random resized crops.

Model	Image Transforms	Loss	Time Steps	Learn. Rate (η)	Decay rate (β)	Accuracy
CSNN	Grayscale	$\mathcal{L}_{CE}$	50	5e-3	0.50	40.79 %
CSNN	Grayscale	$\mathcal{L}_{CE}$	75	5e-3	0.50	41.38 %
CSNN	Grayscale	$\mathcal{L}_{MSE}$	50	5e-3	0.50	47.65 %
CSNN	Grayscale	$\mathcal{L}_{MSE}$	75	5e-3	0.50	47.24 %
CSNN	Bipolar	$\mathcal{L}_{MSE}$	75	5e-3	0.50	58.89 %
VGG9-C	Grayscale	$\mathcal{L}_{MSE}$	75	5e-4	0.75	66.28 %
VGG9-C	Bipolar	$\mathcal{L}_{MSE}$	75	5e-4	0.75	79.37 %
VGG9-C	Bipolar + Data Aug	$\mathcal{L}_{MSE}$	75	5e-4	0.75	81.60 %
VGG9-C	Bipolar + Data Aug	$\mathcal{L}_{MSE}$	100	5e-4	0.75	81.95 %
VGG9-C	Bipolar + Data Aug	$\mathcal{L}_{MSE}$	125	5e-4	0.75	84.22 %
VGG9-F	Grayscale	$\mathcal{L}_{MSE}$	75	5e-4	0.75	66.87 %
VGG9-F	Bipolar	$\mathcal{L}_{MSE}$	75	5e-4	0.75	79.40 %
VGG9-F	Bipolar + Data Aug	$\mathcal{L}_{MSE}$	75	5e-4	0.75	84.11 %
VGG9-F	Bipolar + Data Aug	$\mathcal{L}_{MSE}$	100	5e-4	0.75	85.27 %
VGG9-F	Bipolar + Data Aug	$\mathcal{L}_{MSE}$	125	5e-4	0.75	83.57 %
VGG9-F†	Bipolar + Data Aug	$\mathcal{L}_{MSE}$	75	5e-4	-	82.73 %

On the other hand, considering that SNNs research is still in its infancy, there are very few large scale practical applications [163], but Autonomous Driving (AD) is one of the fields that could benefit the most from SNNs. On the one side, the AD problem space is usually rich in events, suitable for SNNs, on the other, it requires efficient networks for energy-constrained systems. Therefore, some authors have already explored the application of SNNs to AD. For instance, CarSNN, proposed by [164], addresses the problem of car classification from the background in an event-driven dataset (N-CARS Dataset [165]). They resort to Spatio Temporal Back-Propagation (STBP) to train the SOEL [166] system in 3 hierarchical stages for varying sizes of input images. Significantly, the model was implemented on the Loihi Neuromorphic Chip, with a power consumption of only 350 mW, suggesting the potential of SNNs to be deployed in real-time, in resource-constrained systems. Furthermore, [167] proposed a spiking convolutions SNN model with temporal coding for object recognition in LiDAR temporal pulses data.

But more effort is required from the research community to make SNNs a reality in AD. Explicitly, we observe that most works address classification problems, whereas the AD problem space is more complex, usually involving, among other tasks, object detection, object classification, semantic segmentation and panoptic segmentation. Notwithstanding, a few works have already focused on that problem.

[168] have proposed Spiking-YOLO, a converted ANN to SNN that allows object detection with a performance comparable to traditional models. In turn, [169] introduced a SNN strategy for single object localization. Based on the DECOLLE [134] algorithm, a deep local learning scheme, the authors propose an encoder-decoder strategy with 3 layers in each part, to train a spiking convolutions SNN model on the Oxford-IIIT-Pet dataset [170]. They report a mean Intersection over Union (mIoU) of 63.2 % on the test set. However, we can see the limited applicability of these models in real-world scenarios. Moreover, these works resort to convolutional-based strategies, meaning there is still the need of developing fully neuromorphic SNNs that can deal with this kind of complex problems, like AD where many resource constraints are imposed on the models.

VII. CONCLUSION

SNNs are a kind of biologically inspired ANNs. Since information is propagated in the form of spikes, SNNs are supported to be more efficient, particularly if combined with neuromorphic hardware. In fact, most state-of-the-art solutions evidence efficiency gains in SNNs when compared with traditional ANNs. More specifically, directly trained SNNs have demonstrated to be more efficient than converted ANNs to SNNs, although both strategies seem to provide an efficient alternative. However, due to lack of implementation of most suggested SNNs on neuromorphic hardware, it becomes difficult to compare these strategies in terms of efficiency and suitability for such devices, meaning future work would require further validation of some SNN algorithms on neuromorphic hardware.

UL with STDP is one of the most used learning techniques and some models have achieved performance comparable to traditional ANNs, particularly on simple datasets. This is a biologically inspired, hence more efficient, learning method. Notwithstanding, it limits the models to shallow architectures, and it does not work effectively with complex datasets, meaning there is the need to find proper training algorithms. But one key observation from this survey is that it is still not clear how biologically plausible UL could be used to train multilayer SNN models. However, [171] suggest that meta-learning could be helpful for rediscovering biologically plausible synaptic plasticity rules. The authors were able to derive several known local plasticity rules, requiring only the definition of a loss function and the flexible parameterization of the candidate plasticity rules. They argue such a meta-learning approach could lead to novel methodologies for training ANNs. This "learning how to learn" strategy is also biologically plausible as it happened during the millions of years of evolution as well as, for short term survival reasons, it is fundamental during an individual's lifetime. We suggest this could be more extensively studied for SNNs and hypothesise it could help with the unsupervised training of multilayer networks. On the other hand, since the activation functions of SNNs are non-differentiable, proper strategies must be found to allow the supervised training of SNNs. A typical approach is to use surrogate gradients. However, those approximations limit the generalization of the proposed solutions.

Nonetheless, several developments have been made in the last years in the field of SNNs, including learning methods, encoding strategies, connections, and network structures. Many works are inspired by or mimic biological features such as neural circuits of excitation and inhibition, On-Centre and Off-Centre bipolar cells selectivity, expanding RFs, V1 selectivity, etc. But we argue that these mechanisms are still poorly developed in SNNs and more work is necessary towards a unifying strategy to train SNNs and to explore the full extent of known properties of biological neurons in SNNs. Not only could this bring performance gains, but it could also permit fault-tolerant, energy, and data-efficient SNNs.

LIST OF ACRONYMS

AD	Autonomous Driving
ANN	Artificial Neural Network
BN	Batch Normalization
BP	Backpropagation
BPTT	Backpropagation Through Time
C	Causal Set of Input Spikes
CNN	Convolutional Neural Network
DECOLLE	Deep Continuous Local Learning
DL	Deep Learning
DNN	Deep Neural Network
DoG	Difference of Gaussian
DVS	Dynamic Vision Sensor
FB	Feedback
FF	Feedforward
IF	Integrate and Fire
IT	Infrotemporal
LGN	Lateral Geniculate Nucleus
LIF	Leaky Integrate and Fire
LTD	Long Term Depression
LTP	Long Term Potentiation
mIoU	mean Intersection over Union
ML	Machine Learning
MSE(O, Y)	Mean Squared Error
MSE	Mean Squared Error
CE	Cross Entropy
RF	Receptive Field
RGC	Retinal Ganglion Cell
RNN	Recurrent Neural Network
ROC	Rank Order Coding
SL	Supervised Learning
SNN	Spiking Neural Network
SOM	Self Organizing Map
SRM	Spike Response Model
STDP	Spike Timing Dependent Plasticity
RC	Resistor-Capacitor
RL	Reinforcement Learning
SNR	Signal to Noise Ratio
STBP	Spatio Temporal Back-Propagation
T	time

TTFS	Time-To-First-Spike
UL	Unsupervised Learning
W	Synaptic Weight
V1	Primary Visual Area
PLIF	Parametric Leaky Integrate-and-Fire
SEW ResNet	Spike-Element-Wise ResNet
SN	Spiking Neuron
LSM	Liquid State Machine
WTA	Winner-Take-All

ACKNOWLEDGMENT

The authors acknowledge the contributions of all other collaborators of the THEIA project, for their review and critical perspective on this work. Also, they acknowledge the contributions of Tiago Gonçalves and Leonardo Capozzi for their support during the initial stages of this research project.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2012, pp. 1–9. [Online]. Available: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e92%4a68c45b-Paper.pdf

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent., (ICLR) Conf. Track*, 2015, pp. 1–14.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[4] Y. Huang, Y. Cheng, A. Bapna, O. Firat, M. X. Chen, D. Chen, H. J. Lee, J. Ngiam, Q. V. Le, Y. Wu, and Z. Chen, "GPipe: Efficient training of giant neural networks using pipeline parallelism," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, Jun. 2019, pp. 4171–4186.

[6] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, and S. Agarwal, "Language models are few-shot learners," in *Proc. Adv. Neur. Inf. Process. Sys.*, vol. 33, 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8a%c142f64a-Paper.pdf

[7] D. A. Patterson, J. Gonzalez, Q. V. Le, C. Liang, L.-M. Munguía, D. Rothchild, D. R. So, M. Texier, and J. Dean, "Carbon emissions and large neural network training," *ArXiv*, vol. abs/2104.10350, 2021.

[8] N. C. Thompson, K. Greenewald, K. Lee, and G. F. Manso. (Dec. 23, 2021). *Deep Learning's Diminishing Returns*. [Online]. Available: https://spectrum.ieee.org/deep-learning-computational-cost

[9] G. Menghani, "Efficient deep learning: A survey on making deep learning models smaller, faster, and better," *ArXiv*, vol. abs/2106.08962, 2021.

[10] W. Gerstner and W. M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge, U.K.: Cambridge Univ. Press, 2002.

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

[12] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[13] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," *ArXiv*, vol. abs/1708.07747, 2017.

[14] S. Schliebs and N. Kasabov, "Evolving spiking neural network—A survey," *Evolving Syst.*, vol. 4, no. 2, pp. 87–98, 2013, doi: 10.1007/s12530-013-9074-9.

[15] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: Opportunities and challenges," *Frontiers Neurosci.*, vol. 12, p. 774, Oct. 2018, doi: 10.3389/fnins.2018.00774.

[16] D.-A. Nguyen, X.-T. Tran, and F. Iacopi, "A review of algorithms and hardware implementations for spiking neural networks," *J. Low Power Electron. Appl.*, vol. 11, no. 2, p. 23, May 2021. [Online]. Available: https://www.mdpi.com/2079-9268/11/2/23

[17] A. Taherkhani, A. Belatreche, Y. Li, G. Cosma, L. P. Maguire, and T. M. McGinnity, "A review of learning in biologically plausible spiking neural networks," *Neural Netw.*, vol. 122, pp. 253–272, Feb. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608019303181

[18] S. Dora and N. Kasabov, "Spiking neural networks for computational intelligence: An overview," *Big Data Cognit. Comput.*, vol. 5, no. 4, p. 67, Nov. 2021. [Online]. Available: https://www.mdpi.com/2504-2289/5/4/67

[19] W. S. Mcculloch and W. Pitts, "A logical calculus nervous activity," *Bull. Math. Biol.*, vol. 52, no. l, pp. 99–115, 1990.

[20] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1569–1572, Nov. 2003.

[21] S. Dutta, V. Kumar, A. Shukla, N. R. Mohapatra, and U. Ganguly, "Leaky integrate and fire neuron by charge-discharge dynamics in floating-body MOSFET," *Sci. Rep.*, vol. 7, no. 1, p. 8257, Aug. 2017, doi: 10.1038/s41598-017-07418-y.

[22] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Frontiers Neurosci.*, vol. 10, p. 508, Nov. 2016, doi: 10.3389/fnins.2016.00508.

[23] D. O. Hebb, *The Organization of Behavior; A Neuropsychological Theory*, 1st ed. Montreal, QC, Canada: Wiley, 1949.

[24] T. V. P. Bliss and A. R. Gardner-Medwin, "Long-lasting potentiation of synaptic transmission in the dentate area of the unanaesthetized rabbit following stimulation of the perforant path," *J. Physiol.*, vol. 232, no. 2, pp. 357–374, Jul. 1973.

[25] T. Masquelier, R. Guyonneau, and S. J. Thorpe, "Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains," *PLoS ONE*, vol. 3, no. 1, pp. 1–9, Jan. 2008, doi: 10.1371/journal.pone.0001377.

[26] G.-Q. Bi and M.-M. Poo, "Synaptic modifications in cultured hippocampal neurons: Dependence on spike timing, synaptic strength, and postsynaptic cell type," *J. Neurosci.*, vol. 18, no. 24, pp. 10464–10472, Dec. 1998.

[27] G. Chen, H. Cao, J. Conradt, H. Tang, F. Rohrbein, and A. Knoll, "Event-based neuromorphic vision for autonomous driving: A paradigm shift for bio-inspired visual sensing and perception," *IEEE Signal Process. Mag.*, vol. 37, no. 4, pp. 34–49, Jul. 2020.

[28] R. Nelson and V. Connaughton, "Bipolar cell pathways in the vertebrate retina. Different glutamate receptor types for on and off bipolar cells," in *Webvision: The organization of the retina and visual system*. Salt Lake City, UT, USA: Univ. of Utah Health Sciences Center, 2020, pp. 1–48.

[29] F. Bergeaud and S. G. Mallat, "Matching pursuit of images," *Proc. SPIE*, vol. 2491, pp. 2–13, Apr. 1995, doi: 10.1117/12.205366.

[30] L. Paulun, A. Wendt, and N. Kasabov, "A retinotopic spiking neural network system for accurate recognition of moving objects using NeuCube and dynamic vision sensors," *Frontiers Comput. Neurosci.*, vol. 12, p. 42, Jun. 2018, doi: 10.3389/fncom.2018.00042.

[31] R. Vaila, J. Chiasson, and V. Saxena, "Deep convolutional spiking neural networks for image classification," 2019, *arXiv:1903.12272*.

[32] D. Kerr, M. McGinnity, S. Coleman, Q. Wu, and M. Clogenson, "Spiking hierarchical neural network for corner detection," in *Proc. NCTA Int. Conf. Neural Comput. Theory Appl.*, 2011, pp. 230–235.

[33] Q. Fu and H. Dong, "An ensemble unsupervised spiking neural network for objective recognition," *Neurocomputing*, vol. 419, pp. 47–58, Jan. 2021, doi: 10.1016/j.neucom.2020.07.109.

[34] U. S. Kim, O. A. Mahroo, J. D. Mollon, and P. Yu-Wai-Man, "Retinal ganglion cells—Diversity of cell types and clinical relevance," *Frontiers Neurol.*, vol. 12, p. 635, May 2021, doi: 10.3389/fneur.2021.661938.

[35] R. Nelson, "Visual responses of ganglion cells history of electrical recordings," in *Webvision: The Organization of the Retina and Visual System [Internet]*. Salt Lake City, UT, USA: Univ. of Utah Health Sciences Center, 2007, pp. 1–50.

[36] Z. Yu, J. K. Liu, S. Jia, Y. Zhang, Y. Zheng, Y. Tian, and T. Huang, "Toward the next generation of retinal neuroprosthesis: Visual computation with spikes," *Engineering*, vol. 6, no. 4, pp. 449–461, Apr. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2095809920300357

[37] L. Luo, "Architectures of neuronal circuits," *Science*, vol. 373, no. 6559, Sep. 2021, Art. no. eabg7285.

[38] J. S. Isaacson and M. Scanziani, ''How inhibition shapes cortical activity,'' *Neuron*, vol. 72, no. 2, pp. 231–243, Oct. 2011, doi: 10.1016/j.neuron.2011.09.027.

[39] L. Roux and G. Buzsáki, ''Tasks for inhibitory interneurons in intact brain circuits,'' *Neuropharmacology*, vol. 88, pp. 10–23, Jan. 2015, doi: 10.1016/j.neuropharm.2014.09.011.

[40] G. Tian, T. Huang, and S. Wu, ''Excitation-inhibition balanced spiking neural networks for fast information processing,'' in *Proc. IEEE Int. Conf. Syst., Man Cybern. (SMC)*, Oct. 2019, pp. 249–252.

[41] M. V. Tsodyks and T. Sejnowski, ''Rapid state switching in balanced cortical network models,'' *Netw., Comput. Neural Syst.*, vol. 6, no. 2, pp. 111–124, Jan. 1995.

[42] C. van Vreeswijk and H. Sompolinsky, ''Chaos in neuronal networks with balanced excitatory and inhibitory activity,'' *Science*, vol. 274, no. 5293, pp. 1724–1726, Dec. 1996.

[43] M. A. Geramita, S. D. Burton, and N. N. Urban, ''Distinct lateral inhibitory circuits drive parallel processing of sensory information in the mammalian olfactory bulb,'' *eLife*, vol. 5, pp. 1–22, Jun. 2016.

[44] F. Michler, T. Wachtler, and R. Eckhorn, ''Adaptive feedback inhibition improves pattern discrimination learning,'' in *Proc. 2nd Int. Conf. Artif. Neural Netw. Pattern Recognit. (ANNPR)*. Berlin, Germany: Springer, 2006, pp. 21–32, doi: 10.1007/11829898_3.

[45] P. Diehl and M. Cook, ''Unsupervised learning of digit recognition using spike-timing-dependent plasticity,'' *Frontiers Comput. Neurosci.*, vol. 9, p. 99, Aug. 2015, doi: 10.3389/fncom.2015.00099.

[46] D. Zhao, Y. Zeng, and Y. Li, ''BackEISNN: A deep spiking neural network with adaptive self-feedback and balanced excitatory-inhibitory neurons,'' *ArXiv*, vol. abs/2105.13004, 2021.

[47] D. Zhao, Y. Zeng, T. Zhang, M. Shi, and F. Zhao, ''GLSNN: A multi-layer spiking neural network based on global feedback alignment and local STDP plasticity,'' *Frontiers Comput. Neurosci.*, vol. 14, p. 101, Nov. 2020, doi: 10.3389/fncom.2020.576841.

[48] M. Xiao, Q. Meng, Z. Zhang, Y. Wang, and Z. Lin, ''Training feedback spiking neural networks by implicit differentiation on the equilibrium state,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021.

[49] D. Badre, A. Bhandari, H. Keglovits, and A. Kikumoto, ''The dimensionality of neural representations for control,'' *Current Opinion Behav. Sci.*, vol. 38, pp. 20–28, Apr. 2021, doi: 10.1016/j.cobeha.2020.07.002.

[50] B. Babadi and H. Sompolinsky, ''Sparseness and expansion in sensory representations,'' *Neuron*, vol. 83, no. 5, pp. 1213–1226, Sep. 2014, doi: 10.1016/j.neuron.2014.07.035.

[51] M. Farrell, S. Recanatesi, T. Moore, G. Lajoie, and E. Shea-Brown, ''Recurrent neural networks learn robust representations by dynamically balancing compression and expansion,'' *BioRxiv*, pp. 1–20, Jan. 2019.

[52] J. J. M. Reynolds, J. S. Plank, and C. D. Schuman, ''Intelligent reservoir generation for liquid state machines using evolutionary optimization,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.

[53] N. Soures and D. Kudithipudi, ''Deep liquid state machines with neural plasticity for video activity recognition,'' *Frontiers Neurosci.*, vol. 13, p. 686, Jul. 2019, doi: 10.3389/fnins.2019.00686.

[54] Y. Iwashita, A. Takamine, R. Kurazume, and M. S. Ryoo, ''First-person animal activity recognition from egocentric videos,'' in *Proc. 22nd Int. Conf. Pattern Recognit.*, Aug. 2014, pp. 4310–4315.

[55] A. Patiño-Saucedo, H. Rostro-González, T. Serrano-Gotarredona, and B. Linares-Barranco, ''Liquid state machine on SpiNNaker for spatio-temporal classification tasks,'' *Frontiers Neurosci.*, vol. 16, Mar. 2022, Art. no. 819063, doi: 10.3389/fnins.2022.819063.

[56] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, ''Converting static image datasets to spiking neuromorphic datasets using saccades,'' *Frontiers Neurosci.*, vol. 9, p. 437, Nov. 2015, doi: 10.3389/fnins.2015.00437.

[57] S. Wienbar and G. W. Schwartz, ''The dynamic receptive fields of retinal ganglion cells,'' *Prog. Retinal Eye Res.*, vol. 67, pp. 102–117, Nov. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1350946218300284

[58] C. Henley. (2021). *Vision: Central Processing Foundations of Neuroscience*. [Online]. Available: https://openbooks.lib.msu.edu/neuroscience/chapter/vision-central-proce%ssing/

[59] R. J. Love and W. G. Webb, ''Neurosensory organization of speech and hearing,'' in *Neurology for the Speech-Language Pathologist*, 2nd ed., R. J. Love and W. G. Webb, Eds. Oxford, U.K.: Butterworth-Heinemann, 1992, pp. 59–80. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780750690768500113

[60] M. Carandini, ''Area v1,'' *Scholarpedia*, vol. 7, no. 7, 2012, Art. no. 12105.

[61] T. Kadir and M. Brady, ''Saliency, scale and image description,'' *Int. J. Comput. Vis.*, vol. 45, no. 2, pp. 83–105, Nov. 2001.

[62] P. M. López, ''Object component models using Gabor filters for visual recognition,'' Ph.D. dissertation, Dept. Elect. Comput. Eng., Universidade Técnica de Lisboa Instituto Superior Técnico, Lisbon, Portugal, 2008.

[63] R. Panda and B. N. Chatterji, ''Gabor function: An efficient tool for digital image processing,'' *IETE Tech. Rev.*, vol. 13, nos. 4–5, pp. 225–231, Jul. 1996.

[64] P. N. Loxley, ''The two-dimensional Gabor function adapted to natural image statistics: A model of simple-cell receptive fields and sparse structure in images,'' *Neural Comput.*, vol. 29, no. 10, pp. 2769–2799, Oct. 2017, doi: 10.1162/NECO_a_00997.

[65] T. Serre, *Hierarchical Models of the Visual System*. New York, NY, USA: Springer, 2013, pp. 1–12, doi: 10.1007/978-1-4614-7320-6_345-1.

[66] A. Kanazawa, A. Sharma, and D. Jacobs, ''Locally scale-invariant convolutional neural networks,'' 2014, *arXiv:1412.5104*.

[67] O. S. Kayhan and J. C. van Gemert, ''On translation invariance in CNNs: Convolutional layers can exploit absolute spatial location,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 14262–14273.

[68] G. Nam, H. Choi, J. Cho, and I.-J. Kim, ''PSI-CNN: A pyramid-based scale-invariant CNN architecture for face recognition robust to various image resolutions,'' *Appl. Sci.*, vol. 8, no. 9, p. 1561, Sep. 2018.

[69] H. Nasrallah, M. Shukor, and A. J. Ghandour, ''Sci-Net: A scale invariant model for buildings segmentation from aerial images,'' 2021, *arXiv:2111.06812*.

[70] D. Scherer, A. Müller, and S. Behnke, ''Evaluation of pooling operations in convolutional architectures for object recognition,'' in *Artificial Neural Networks—ICANN 2010*, K. Diamantaras, W. Duch, and L. S. Iliadis, Eds. Berlin, Germany: Springer, 2010, pp. 92–101.

[71] J. Liu, H. Huo, W. Hu, and T. Fang, ''Brain-inspired hierarchical spiking neural network using unsupervised STDP rule for image classification,'' in *Proc. 10th Int. Conf. Mach. Learn. Comput.*, Feb. 2018, pp. 230–235.

[72] S. R. Kheradpisheh, M. Ganjtabesh, and T. Masquelier, ''Bio-inspired unsupervised learning of visual features leads to robust invariant object recognition,'' *Neurocomputing*, vol. 205, pp. 382–392, Sep. 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231216302880

[73] T. Masquelier and S. J. Thorpe, ''Unsupervised learning of visual features through spike timing dependent plasticity,'' *PLoS Comput. Biol.*, vol. 3, no. 2, pp. 247–257, 2007.

[74] A. Tavanaei, T. Masquelier, and A. S. Maida, ''Acquisition of visual features through probabilistic spike-timing-dependent plasticity,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 307–314.

[75] B. Zhao, R. Ding, S. Chen, B. Linares-Barranco, and H. Tang, ''Feedforward categorization on AER motion events using cortex-like features in a spiking neural network,'' *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 9, pp. 1963–1978, Sep. 2015.

[76] S. R. Kheradpisheha, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, ''STDP-based spiking deep convolutional neural networks for object recognition,'' *Neural Netw.*, vol. 99, pp. 56–67, Mar. 2018, doi: 10.1016/j.neunet.2017.12.005.

[77] B. Illing, W. Gerstner, and J. Brea, ''Biologically plausible deep learning—But how far can we go with shallow networks?'' *Neural Netw.*, vol. 118, pp. 90–101, Oct. 2019, doi: 10.1016/j.neunet.2019.06.001.

[78] D. Auge, J. Hille, E. Mueller, and A. Knoll, ''A survey of encoding techniques for signal processing in spiking neural networks,'' *Neural Process. Lett.*, vol. 53, no. 6, pp. 4693–4710, Dec. 2021.

[79] E. D. Adrian and Y. Zotterman, ''The impulses produced by sensory nerve endings: Part 3. Impulses set up by touch and pressure,'' *J. Physiol.*, vol. 61, no. 4, pp. 465–483, Aug. 1926.

[80] R. M. Enoka and J. Duchateau, ''Rate coding and the control of muscle force,'' *Cold Spring Harbor Perspect. Med.*, vol. 7, no. 10, Oct. 2017, Art. no. a029702.

[81] Y. Wang, Y. Zeng, J. Tang, and B. Xu, ''Biological neuron coding inspired binary word embeddings,'' *Cognit. Comput.*, vol. 11, no. 5, pp. 676–684, Oct. 2019.

[82] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[83] N. H. Salminen, P. J. C. May, P. Alku, and H. Tiitinen, ''A population rate code of auditory space in the human cortex,'' *PLoS ONE*, vol. 4, no. 10, Oct. 2009, Art. no. e7600.

[84] A. Ihlefeld, N. Alamatsaz, and R. M. Shapley, ''Population rate-coding predicts correctly that human sound localization depends on sound intensity,'' *eLife*, vol. 8, Oct. 2019, Art. no. e47027.

[85] A. Pasupathy and C. E. Connor, ''Population coding of shape in area V4,'' *Nature Neurosci.*, vol. 5, no. 12, pp. 1332–1338, Nov. 2002.

[86] J. S. Montijn, M. Vinck, and C. M. A. Pennartz, ''Population coding in mouse visual cortex: Response reliability and dissociability of stimulus tuning and noise correlation,'' *Frontiers Comput. Neurosci.*, vol. 8, p. 58, Jun. 2014.

[87] S. J. Thorpe, D. Fize, and C. Marlot, ''Speed of processing in the human visual system,'' *Nature*, vol. 381, no. 6582, pp. 520–522, 1996.

[88] T. Gollisch and M. Meister, ''Rapid neural coding in the retina with relative spike latencies,'' *Science*, vol. 319, no. 5866, pp. 1108–1111, Feb. 2008.

[89] E. T. Rolls, L. Franco, N. C. Aggelopoulos, and J. M. Jerez, ''Information in the first spike, the order of spikes, and the number of spikes provided by neurons in the inferior temporal visual cortex,'' *Vis. Res.*, vol. 46, no. 25, pp. 4193–4205, Nov. 2006.

[90] A. Resulaj, S. Ruediger, S. R. Olsen, and M. Scanziani, ''First spikes in visual cortex enable perceptual discrimination,'' *eLife*, vol. 7, Apr. 2018, Art. no. e34044.

[91] G. Portelli, J. M. Barrett, G. Hilgen, T. Masquelier, A. Maccione, S. D. Marco, L. Berdondini, P. Kornprobst, and E. Sernagor, ''Rank order coding: A retinal information decoding strategy revealed by large-scale multielectrode array retinal recordings,'' *Eneuro*, vol. 3, no. 3, pp. 1–18, May 2016.

[92] M. A. Montemurro, M. J. Rasch, Y. Murayama, N. K. Logothetis, and S. Panzeri, ''Phase-of-firing coding of natural visual stimuli in primary visual cortex,'' *Current Biol.*, vol. 18, no. 5, pp. 375–380, Mar. 2008.

[93] H. K. Turesson, N. K. Logothetis, and K. L. Hoffman, ''Category-selective phase coding in the superior temporal sulcus,'' *Proc. Nat. Acad. Sci. USA*, vol. 109, no. 47, pp. 19438–19443, Nov. 2012.

[94] A. Cattani, G. T. Einevoll, and S. Panzeri, ''Phase-of-firing code,'' 2015, *arXiv:1504.03954*.

[95] F. Zeldenrust, W. J. Wadman, and B. Englitz, ''Neural coding with bursts—Current state and future perspectives,'' *Frontiers Comput. Neurosci.*, vol. 12, p. 48, Jul. 2018.

[96] S. Park, S. Kim, B. Na, and S. Yoon, ''T2FSNN: Deep spiking neural networks with time-to-first-spike coding,'' in *Proc. 57th ACM/IEEE Design Autom. Conf. (DAC)*, Jul. 2020, pp. 1–6.

[97] S. Park, S. Kim, H. Choe, and S. Yoon, ''Fast and efficient information transmission with burst spikes in deep spiking neural networks,'' in *Proc. 56th Annu. Design Autom. Conf.*, Jun. 2019, pp. 1–6.

[98] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, ''Learning representations by back-propagating errors,'' *Nature*, vol. 323, pp. 533–536, Oct. 1986.

[99] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, ''Greedy layer-wise training of deep networks,'' in *Proc. 19th Int. Conf. Neural Inf. Process. Syst. (NIPS)*, Cambridge, MA, USA: MIT Press, 2006, pp. 153–160.

[100] G. E. Hinton, ''What kind of a graphical model is the brain?'' in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2005, pp. 1765–1775.

[101] G. Hinton, S. Osindero, and Y. Teh, ''A fast learning algorithm for deep belief nets,'' *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006, doi: 10.1162/neco.2006.18.7.1527.

[102] C. Poultney, S. Chopra, Y. L. Cun, and Others, ''Efficient learning of sparse representations with an energy-based model,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 1137–1144.

[103] Y. LeCun, Y. Bengio, and G. Hinton, ''Deep learning,'' *Nature*, vol. 521, pp. 436–444, Feb. 2015.

[104] Y. Bengio, D.-H. Lee, J. Bornschein, T. Mesnard, and Z. Lin, ''Towards biologically plausible deep learning,'' 2015, *arXiv:1502.04156*.

[105] T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman, ''Random synaptic feedback weights support error backpropagation for deep learning,'' *Nature Commun.*, vol. 7, no. 1, pp. 1–10, Dec. 2016, doi: 10.1038/ncomms13276.

[106] M. R. Carey, J. F. Medina, and S. G. Lisberger, ''Instructive signals for motor learning from visual cortical area MT,'' *Nature Neurosci.*, vol. 8, no. 6, pp. 813–819, Jun. 2005.

[107] K. Doya, ''What are the computations of the cerebellum, the basal ganglia and the cerebral cortex?'' *Neural Netw.*, vol. 12, nos. 7–8, pp. 961–974, 1999.

[108] M. Ito, ''Mechanisms of motor learning in the cerebellum,'' *Brain Res.*, vol. 886, nos. 1–2, pp. 237–245, Dec. 2000.

[109] E. I. Knudsen, ''Instructed learning in the auditory localization pathway of the barn owl,'' *Nature*, vol. 417, no. 6886, pp. 322–328, May 2002.

[110] I. Sporea and A. Grüning, ''Supervised learning in multilayer spiking neural networks,'' *Neural Comput.*, vol. 25, no. 2, pp. 473–509, Feb. 2013.

[111] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer, ''Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2015, pp. 1–8.

[112] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, ''Conversion of continuous-valued deep networks to efficient event-driven networks for image classification,'' *Frontiers Neurosci.*, vol. 11, 2017, doi: 10.3389/fnins.2017.00682.

[113] S.-W. Deng and S. Gu, ''Optimal conversion of conventional artificial neural networks to spiking neural networks,'' *ArXiv*, vol. abs/2103.00476, 2021.

[114] J. Ding, Z. Yu, Y. Tian, and T. Huang, ''Optimal ANN-SNN conversion for fast and accurate inference in deep spiking neural networks,'' in *Proc. 13th Int. Joint Conf. Artif. Intell.*, Aug. 2021, pp. 2328–2336, doi: 10.24963/ijcai.2021/321.

[115] B. Rueckauer and S.-C. Liu, ''Conversion of analog to spiking neural networks using sparse temporal coding,'' in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2018, pp. 1–5.

[116] S. Shrestha and G. Orchard, ''SLAYER: Spike layer error reassignment in time,'' in *Proc. NeurIPS*, 2018, pp. 1–10.

[117] D. Shulz and D. Feldman, ''Spike timing-dependent plasticity,'' in *Neural Circuit Development and Function in the Brain*, J. L. Rubenstein and P. Rakic, Eds. Oxford, U.K.: Academic, 2013, pp. 155–181. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780123972675000297

[118] A. Morrison, A. Aertsen, and M. Diesmann, ''Spike-timing-dependent plasticity in balanced random networks,'' *Neural Comput.*, vol. 19, no. 6, pp. 1437–1467, Jun. 2007.

[119] B. Nessler, M. Pfeiffer, L. Buesing, and W. Maass, ''Bayesian computation emerges in generic cortical microcircuits through spike-timing-dependent plasticity,'' *PLOS Comput. Biol.*, vol. 9, no. 4, pp. 1–30, Apr. 2013, doi: 10.1371/journal.pcbi.1003037.

[120] D. Querlioz, O. Bichler, P. Dollfus, and C. Gamrat, ''Immunity to device variations in a spiking neural network with memristive nanodevices,'' *IEEE Trans. Nanotechnol.*, vol. 12, no. 3, pp. 288–295, May 2013.

[121] J.-P. Pfister and W. Gerstner, ''Triplets of spikes in a model of spike timing-dependent plasticity,'' *J. Neurosci.*, vol. 26, no. 38, pp. 9673–9682, Sep. 2006.

[122] H. Hazan, D. Saunders, D. T. Sanghavi, H. Siegelmann, and R. Kozma, ''Unsupervised learning with self-organizing spiking neural networks,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–6.

[123] T. Kohonen, ''Self-organized formation of topologically correct feature maps,'' *Biol. Cybern.*, vol. 43, pp. 59–69, Jan. 1982.

[124] J. Brader, W. Senn, and S. Fusi, ''Learning real-world stimuli in a neural network with spike-driven synaptic dynamics,'' *Neural Comput.*, vol. 19, no. 11, pp. 2881–2912, 2007.

[125] G. Srinivasan, P. Panda, and K. Roy, ''STDP-based unsupervised feature learning using convolution-over-time in spiking neural networks for energy-efficient neuromorphic computing,'' *ACM J. Emerg. Technol. Comput. Syst.*, vol. 14, no. 4, pp. 1–12, Oct. 2018.

[126] A. Shrestha, K. Ahmed, Y. Wang, and Q. Qiu, ''Stable spike-timing dependent plasticity rule for multilayer unsupervised and supervised learning,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 1999–2006.

[127] P. Falez, P. Tirilly, I. Marius Bilasco, P. Devienne, and P. Boulet, ''Multi-layered spiking neural network with target timestamp threshold adaptation and STDP,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.

[128] A. Tavanaei and A. S. Maida, ''Multi-layer unsupervised learning in a spiking convolutional neural network,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 2023–2030.

[129] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, ''ImageNet: A large-scale hierarchical image database,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[130] C. Lee, S. S. Sarwar, P. Panda, G. Srinivasan, and K. Roy, ''Enabling spike-based backpropagation for training deep neural network architectures,'' *Frontiers Neurosci.*, vol. 14, pp. 119, Feb. 2020, doi: 10.3389/fnins.2020.00119.

[131] H. Mostafa, "Supervised learning based on temporal coding in spiking neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 3227–3235, Jul. 2017.

[132] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J. M. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proc. IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

[133] N. Qiao, H. Mostafa, F. Corradi, M. Osswald, F. Stefanini, D. Sumislawska, and G. Indiveri, "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128K synapses," *Frontiers Neurosci.*, vol. 9, p. 141, Apr. 2015, doi: 10.3389/fnins.2015.00141.

[134] J. Kaiser, H. Mostafa, and E. Neftci, "Synaptic plasticity dynamics for deep continuous local learning (DECOLLE)," *Frontiers Neurosci.*, vol. 14, p. 424, May 2020, doi: 10.3389/fnins.2020.00424.

[135] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. D. Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, J. Kusnitz, M. Debole, S. Esser, T. Delbruck, M. Flickner, and D. Modha, "A low power, fully event-based gesture recognition system," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7388–7397.

[136] J. Göltz, L. Kriener, A. Baumbach, S. Billaudelle, O. Breitwieser, B. Cramer, D. Dold, A. F. Kungl, W. Senn, J. Schemmel, K. Meier, and M. A. Petrovici, "Fast and energy-efficient neuromorphic deep learning with first-spike times," *Nature Mach. Intell.*, vol. 3, no. 9, pp. 823–835, Sep. 2021.

[137] I. M. Comsa, K. Potempa, L. Versari, T. Fischbacher, A. Gesmundo, and J. Alakuijala, "Temporal coding in spiking neural networks with alpha synaptic function," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 8529–8533.

[138] D. Sterratt, B. Graham, A. Gillies, and D. Willshaw, "The synapse," in *Principles of Computational Modelling in Neuroscience*. Cambridge, U.K.: Cambridge Univ. Press, 2011, pp. 172–195. [Online]. Available: https://www.cambridge.org/core/product/identifier/CBO9780511975899A063/%type/book_part

[139] S. Zhou, X. Li, Y. Chen, S. T. Chandrasekaran, and A. Sanyal, "Temporal-coded deep spiking neural network with easy training and robust performance," *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 12, May 2021, pp. 11143–11151. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/17329

[140] S. R. Kheradpisheh and T. Masquelier, "Temporal backpropagation for spiking neural networks with one spike per neuron," *Int. J. Neural Syst.*, vol. 30, no. 6, Jun. 2020, Art. no. 2050027, doi: 10.1142/S0129065720500276.

[141] T. C. Wunderlich and C. Pehle, "Event-based backpropagation can compute exact gradients for spiking neural networks," *Sci. Rep.*, vol. 11, no. 1, pp. 1–17, Dec. 2021, doi: 10.1038/s41598-021-91786-z.

[142] Y. LeCun, "A theoretical framework for back-propagation," 1988.

[143] W. Fang, Z. Yu, Y. Chen, T. Masquelier, T. Huang, and Y. Tian, "Incorporating learnable membrane time constant to enhance learning of spiking neural networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 2641–2651.

[144] W. Cheng, H. Luo, W. Yang, L. Yu, and W. Li, "Structure-aware network for lane marker extraction with dynamic vision sensor," 2020, *arXiv:2008.06204*.

[145] N. Rathi and K. Roy, "DIET-SNN: A low-latency spiking neural network with direct input encoding and leakage and threshold optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Oct. 1, 2021, doi: 10.1109/TNNLS.2021.3111897.

[146] Y. Hu, H. Tang, and G. Pan, "Spiking deep residual networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Nov. 1, 2021, doi: 10.1109/TNNLS.2021.3119238.

[147] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, "Going deeper in spiking neural networks: VGG and residual architectures," *Frontiers Neurosci.*, vol. 13, p. 95, Mar. 2019, doi: 10.3389/fnins.2019.00095.

[148] W. Fang, Z. Yu, Y. Chen, T. Huang, T. Masquelier, and Y. Tian, "Deep residual learning in spiking neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds. Red Hook, NY, USA: Curran Associates, 2021, pp. 21056–21069. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/afe434653a898da200440412%62b3ac74-Paper.pdf

[149] M. Yuan, X. Wu, R. Yan, and H. Tang, "Reinforcement learning in spiking neural networks with stochastic and deterministic synapses," *Neural Comput.*, vol. 31, no. 12, pp. 2368–2389, Dec. 2019, doi: 10.1162/neco_a_01238.

[150] R. V. Florian, "A reinforcement learning algorithm for spiking neural networks," in *Proc. 7th Int. Symp. Symbolic Numeric Algorithms Sci. Comput. (SYNASC)*, Sep. 2005, p. 8.

[151] R. Bogacz, "Dopamine role in learning and action inference," *eLife*, vol. 9, pp. 1–33, Jul. 2020.

[152] S. B. Flagel, J. J. Clark, T. E. Robinson, L. Mayo, A. Czuj, C. A. Akers, S. M. Clinton, P. E. M. Phillips, and H. Akil, "A selective role for dopamine in reward learning," *Nature*, vol. 469, no. 7328, pp. 53–57, 2011.

[153] R. A. Wise, "Dopamine, learning and motivation," *Nature Rev. Neurosci.*, vol. 5, no. 6, pp. 483–494, Jun. 2004.

[154] S. Deng and S. Gu, "Optimal conversion of conventional artificial neural networks to spiking neural networks," 2021, *arXiv:2103.00476*.

[155] A. Tavanaei and A. Maida, "BP-STDP: Approximating backpropagation using spike timing dependent plasticity," *Neurocomputing*, vol. 330, pp. 39–47, Feb. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231218313420

[156] H. Hazan, D. J. Saunders, H. Khan, D. Patel, D. T. Sanghavi, H. T. Siegelmann, and R. Kozma, "BindsNET: A machine learning-oriented spiking neural networks library in Python," *Frontiers Neuroinform.*, vol. 12, p. 89, Dec. 2018, doi: 10.3389/fninf.2018.00089.

[157] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, Jun. 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[158] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. NeurIPS*, 2019.

[159] D. Goodman, "Brian: A simulator for spiking neural networks in Python," *Frontiers Neuroinform.*, vol. 2, p. 5, Nov. 2008, doi: 10.3389/neuro.11.005.2008.

[160] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, Feb. 2012.

[161] J. K. Eshraghian, M. Ward, E. Neftci, X. Wang, G. Lenz, G. Dwivedi, M. Bennamoun, D. S. Jeong, and W. D. Lu, "Training spiking neural networks using lessons from deep learning," 2021, *arXiv:2109.12894*.

[162] F. Zenke and S. Ganguli, "SuperSpike: Supervised learning in multilayer spiking neural networks," *Neural Comput.*, vol. 30, no. 6, pp. 1514–1541, Jun. 2018, doi: 10.1162/neco_a_01086.

[163] S. Mohapatra, H. Gotzig, S. Yogamani, S. Milz, and R. Zöllner, "Exploring deep spiking neural networks for automated driving applications," in *Proc. 14th Int. Joint Conf. Comput. Vis., Imag. Comput. Graph. Theory Appl.*, Jan. 2019, pp. 1–8.

[164] A. Viale, A. Marchisio, M. Martina, G. Masera, and M. Shafique, "CarSNN: An efficient spiking neural network for event-based autonomous cars on the Loihi neuromorphic research processor," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–10.

[165] A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman, "HATS: Histograms of averaged time surfaces for robust event-based object classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1731–1740.

[166] K. Stewart, G. Orchard, S. B. Shrestha, and E. Neftci, "Online few-shot gesture learning on a neuromorphic processor," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 10, no. 4, pp. 512–521, Dec. 2020.

[167] W. Wang, S. Zhou, J. Li, X. Li, J. Yuan, and Z. Jin, "Temporal pulses driven spiking neural network for time and power efficient object recognition in autonomous driving," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 6359–6366.

[168] S. Kim, S. Park, B. Na, and S. Yoon, "Spiking-YOLO: Spiking neural network for energy-efficient object detection," 2019, *arXiv:1903.06530*.

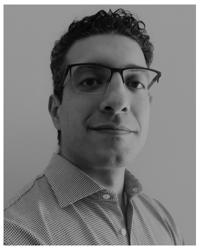
[169] S. Barchid, J. Mennesson, and C. Djeraba, "Deep spiking convolutional neural network for single object localization based on deep continuous local learning," in *Proc. Int. Conf. Content-Based Multimedia Indexing (CBMI)*, Jun. 2021, pp. 1–5.

[170] O. M Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, "Cats and dogs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3498–3505.

[171] B. Confavreux, F. Zenke, E. Agnes, T. Lillicrap, and T. Vogels, "A meta-learning approach to (re)discover plasticity rules that carve a desired function into a neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds. Red Hook, NY, USA: Curran Associates, 2020, pp. 16398–16408. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/bdbd5ebfde4934142c8a88e7%a3796cd5-Paper.pdf

JOÃO D. NUNES received the M.Sc. degree in biomedical engineering from the NOVA School of Science and Technology, Lisbon, Portugal, in 2021, with a thesis on the neural correlates of motor imagery and motor observation tasks. During his studies, he took courses with a special focus on biomedical signals and image processing, electronics, information systems, and machine learning. He now collaborates in the THEIA project, as a Research Assistant, with the Faculty of Engineering, University of Porto, Porto, Portugal, where he works on efficient deep learning algorithms with a particular focus on bioinspired spiking neural networks. He is also a Research Assistant with INESC TEC, Porto, where he investigates deep learning algorithms to detect and segment nuclei from histopathology microscopic imaging. His other research interests include machine learning, computer vision, pattern recognition, and the medical imaging domain.

MARCELO CARVALHO was born in Paredes, Porto, Portugal, in 1999. He is currently pursuing the M.Sc. degree in biomedical engineering with the Faculty of Engineering, University of Porto, Porto. He is developing his master thesis on neuromorphic architectures for efficient perception, exploring spiking neural network potentialities, in collaboration with the THEIA project, with the Faculty of Engineering, University of Porto. In 2021, he was a member of the Biometric Research Group with the goal of developing deep learning methods to detect facial presentation attacks, and, in the same year, he was also involved in the IEEE VIP Cup 2021 for in-bed pose estimation. His research interests include machine learning and computer vision fields with particular emphasis on deep learning for biomedical applications.



DIOGO CARNEIRO received the M.Sc. degree in electronic and telecommunications engineering from Aveiro University, Aveiro, Portugal, in 2017, with a thesis on generalization and anticipation skills on robot ball catching using supervised learning, where it was explored the anticipation of ball trajectories based on human motion, as well as the generalization of robot motion based on dynamic movement primitives. He has been working with Bosch Car Multimedia Portugal, S.A., Braga, Portugal, since 2018, as a Deep Learning and Machine Learning Researcher in several areas such as computer vision and digital signal processing in the context of interior vehicle sensing and autonomous driving. His research interests include deep learning areas with practical effects on real-world applications.

JAIME S. CARDOSO (Senior Member, IEEE) is a Full Professor with the Faculty of Engineering of the University of Porto (FEUP). From 2012 to 2015, he was the President of the Portuguese Association for Pattern Recognition (APRP), affiliated with the IAPR. He has coauthored 300+ papers, 100+ of which in international journals, which attracted 7000+ citations, according to google scholar. His research interests include computer vision, machine learning, and decision support systems. Image and video processing focuses on medicine and biometrics. The work on machine learning cares mostly with the adaptation of learning to the challenging conditions presented by visual data, with a focus on deep learning and explainable machine learning. The particular emphasis of the work in decision support systems goes to medical applications, always anchored on the automatic analysis of visual data.