TU Delft

Delft University of Technology

A Reconfigurable Graphene-Based Spiking Neural Network Architecture

Wang, He; Cucu Laurenciu, Nicoleta; Cotofana, Sorin Dan

DOI
10.1109/OJNANO.2021.3094761

Publication date
2021

Document Version
Final published version

Published in
IEEE Open Journal of Nanotechnology

Citation (APA)
Wang, H., Cucu Laurenciu, N., & Cotofana, S. D. (2021). A Reconfigurable Graphene-Based Spiking Neural Network Architecture. *IEEE Open Journal of Nanotechnology*, *2*, 59-71. Article 9477033. https://doi.org/10.1109/OJNANO.2021.3094761

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

IEEE Open Journal of
Nanotechnology

Received 18 April 2021; revised 18 June 2021; accepted 1 July 2021. Date of publication 7 July 2021;
date of current version 21 July 2021. The review of this article was arranged by Associate Editor Masaki Tanemura.

Digital Object Identifier 10.1109/OJNANO.2021.3094761

A Reconfigurable Graphene-Based Spiking Neural Network Architecture

HE WANG [1] (Student Member, IEEE), NICOLETA CUCU LAURENCIU [1] (Member, IEEE), AND SORIN DAN COTOFANA [1] (Fellow, IEEE)

[1] Department of Quantum and Computer Engineering, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 Delft CD, The Netherlands

CORRESPONDING AUTHOR: HE WANG (e-mail: H.Wang-13@tudelft.nl).

ABSTRACT In the paper we propose a reconfigurable graphene-based Spiking Neural Network (SNN) architecture and a training methodology for initial synaptic weight values determination. The proposed graphene-based platform is flexible, comprising a programmable synaptic array which can be configured for different initial synaptic weights and plasticity functionalities and a spiking neuronal array, onto which various neural network structures can be mapped according to the application requirements and constraints. To demonstrate the validity of the synaptic weights training methodology and the suitability of the proposed SNN architecture for practical utilization, we consider character recognition and edge detection applications. In each case, the graphene-based platform is configured as per the application tailored SNN topology and initial state and SPICE simulated to evaluate its reaction to the applied input stimuli. For the first application, a 2-layer SNN is used to perform character recognition for 5 vowels. Our simulation indicates that the graphene-based SNN can achieve comparable recognition accuracy with the one delivered by a functionally equivalent Artificial Neural Network. Further, we reconfigure the architecture for a 3-layer SNN to perform edge detection on 2 grayscale images. SPICE simulation results indicate that the edge extraction results are close agreement with the one produced by classical edge detection operators. Our results suggest the feasibility and flexibility of the proposed approach for various application purposes. Moreover, the utilized graphene-based synapses and neurons operate at low supply voltage, consume low energy per spike, and exhibit small footprints, which are desired properties for largescale energy-efficient implementations.

INDEX TERMS Spiking neural network, graphene, reconfigurable, character recognition, edge detection.

I. INTRODUCTION

Neuromorphic computing research [1], [2] is receiving massive attention as it helps to investigate the essential functionality underneath the human brain's attractive properties (e.g., robustness, highly parallel information processing capabilities, energy effectiveness, handling complex tasks suitability) and promote the design and implementation of bio-inspired systems with brain-akin computation abilities. Spiking Neural Networks (SNNs) are of particular interest for neuromorphic designs, as they capture biologically plausible neural network dynamics and exhibit potential for increased energy efficiency desirable for building large-scale computation systems [3], [4]. In an SNN, spiking neurons are the fundamental information processing units and synapses the junctions connecting them, and as such, the key challenge for the realization of any brain-inspired computation system is the design of artificial synapses and neurons that are suitable for large scale implementations.

Nowadays, most neuromorphic systems, e.g., [5]–[7], are implemented with CMOS technology by using complex CMOS circuitry to mimic synapse and neuron functionalities. These systems suffer from high energy consumption and limited scalability. More recently, various emerging technologies are also utilized in neuromorphic systems, e.g., memristor [8] and phase change memory [9], as they exhibit simple structure and electronic properties desirable for SNN emulation. However, these technologies are mainly used to simplify the designs of individual synapses and neurons [10]–[13], while

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

their usage in complete SNN hardware implementations still require additional CMOS circuitry to enable the basic SNN functionality [14]–[17], which is restricting their performance for large-scale energy-efficient implementations.

Graphene is a promising post-Si candidate with outstanding properties [18], [19], e.g., ballistic transport, ultimate thinness, flexibility. Moreover, when compared with other emerging technologies, graphene is a biocompatible material, which makes it favourable for graphene-based neuromorphic biointerfaces. Previous work demonstrated graphene's suitability for artificial synapse [20], [21], neuron [22], and SNN unit implementations [23], thus a versatile, generic graphene-based SNN architecture that can be reconfigured for various practical tasks, would facilitate the exploration of grahene-based neuromorphic computing capabilities.

In this paper, we propose a reconfigurable graphene-based SNN architecture and an associated training methodology for initial synaptic weight values determination, which main features can be summarized as: (i) area and energy efficiency due to effective graphene-based implementation of neurons and synapses, (ii) flexible support for SNN applications mapping due to FPGA-alike reconfiguration feature, and (iii) training process simplicity due to Spike-Timing-Dependent Plasticity (STDP) and Long-Term Plasticity (LTP) support. Specifically, the reconfigurable SNN architecture comprises a synaptic array (consisting of graphene-based programmable synapses) and a neuronal array (consisting of graphene-based spiking neurons), onto which various network structures can be mapped for different application scenarios. Furthermore, the synapses can be configured for different initial synaptic weights and plasticity, e.g., Long-Term Potentiation (LTP) and Long-Term Depression (LTD). To reconfigure the proposed graphene-based platform for a practical application, two ingredients are required: an SNN topology and an initial SNN state, e.g., initial synaptic weights. The general flow for using the proposed reconfigurable platform for real-life scenarios is as follows: For a given application and SNN topology, the SNN initial synaptic weight values are first determined by means of the specific training method described in Section III-B. Subsequently, the SNN topology is mapped onto the graphene-based SNN architecture by establishing the configuration of the programmable interconnect matrix, and the synaptic array initial state (synaptic weight values and plasticity types).

To investigate the versatility and suitability of the proposed reconfigurable architecture for practical applications, we consider 2 SNN topologies tailored for character recognition and edge detection, map them on the proposed graphene neuroplatform, and evaluate their performance by means of SPICE simulations. For the first application, a 2-layer SNN consisting of 30 neurons is utilized to recognize the vowel characters, i.e., "A," "E," "I," "O," and "U" (and their variations), represented by 5×5 black and white pixel matrices. The response of the graphene-based SNN architecture, reconfigured according to the considered SNN topology and different initial synaptic weight values is evaluated for multiple input datasets comprising the original characters and their variations, is evaluated by means of SPICE simulations. The obtained results indicate that a recognition accuracy of up to 94.5% is achieved, which is in line (maximum 7.8% deviation) with the one obtained by means of Matlab simulation of a functionally equivalent Artificial Neural Network (ANN).

For the second application, we consider a 3-layer SNN consisting of 13 neurons for performing edge detection on 2 images, i.e., Lena and Cameraman. To this end, for each and every image pixel, we should determine whether it belongs to an edge or not and this can be done by sequentially analyzing the pixel configuration of the 3×3 grayscale pixel matrix centered around it. To obtain the SNN initial synaptic weights, we make use of a set of directional edge and non-edge 3×3 kernels. The graphene-based SNN architecture is configured according to the SNN topology and initial state, and then SPICE simulations of all possible 3×3 pixel matrix instances are performed to obtain the edge extracted output image. Simulation results reveal that the graphene-based SNN platform delivers comparable results when compared with to one produced by classical edge detectors, i.e., Canny, Roberts, Sobel, Prewitt [24], which suggests good perceptual edge extracted image quality. If Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE) are utilized as evaluation metrics the SNN approach delivers slightly worse PSNR and MSE figures, i.e., 2.3% lower PSNR, and 3% higher MSE, for Lena, while for the Cameraman it marginally outperforms the classical edge detectors by 2.7% for PSNR and 2.9% for MSE.

The simulation results demonstrate that the proposed SNN platform is able to properly perform character recognition and edge detection tasks, which suggests the feasibility and flexibility of the proposed approach for various application purposes. Moreover, the utilized graphene-based synapses and neurons operate at low supply voltage (200 mV), consumes low energy per spike for both neuron (43pJ and 5.2×10^{-7}pJ at 200Hz and 20GHz spike frequency scale, respectively) and synapse (5.1pJ and 6.0×10^{-8}pJ at 200Hz and 20GHz spike frequency scale, respectively), and a graphene-based synapse occupies an active area of $\approx$45nm^2 (2 GNR devices) and a neuron an active area of $\approx$176nm^2 (6 GNR devices), which are desired properties for large-scale energy-efficient implementations.

The remaining of this paper is organized as follows: In Section II we describe the SNN functionality, the graphene-based SNN unit, and present a general view of the simulation framework. Section III introduces the proposed reconfigurable graphene-based SNN architecture and the associated training methodology for deriving the initial synaptic weight values. Section IV presents the simulation results for character recognition and edge detection applications, while Section V concludes the paper.

II. BACKGROUND

In this section, we introduce the basic structure and functionality of a Spiking Neural Network (SNN), present the generic

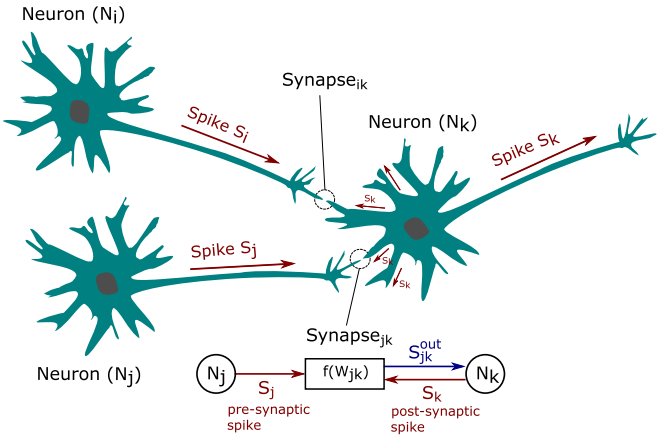


FIGURE 1. Spiking neural network illustration.

graphene-based SNN unit, which constitutes the fundamental building block for our proposal, and conclude by providing a brief account on the utilized graphene circuit SPICE simulation framework.

A. SPIKING NEURAL NETWORKS

SNNs comprise of spiking neurons, as basic information processing units and synapses, as junctions interconnecting the neurons [3]. A distinctive characteristic of SNNs is that the internal signal transmission is carried out via electric spikes. Fig. 1 depicts a small SNN example consisting of 3 spiking neurons (N_i, N_j, and N_k) that are connected via 2 synapses ($\text{Synapse}_{\text{ik}}$ and $\text{Synapse}_{\text{jk}}$). Neuron N_k collects input spikes (S_i and S_j) from the other neurons, and generates an output spike S_k when the cumulated input reaches its firing threshold. Immediately after the firing event, neuron N_k enters a refractory period, during which it doesn't react to incoming spikes. There are various spiking neuron models to describe a neuron's functionality [25], [26], among which the standard non-linear Leaky Integrate and Fire (LIF) is of particular interest, as it captures the essential behavior of a spiking neuron while having a relatively low complexity [3].

Synapses, while essentially known as signal transmission media between adjacent neurons, assume a processing role as well since their transmission efficiency (synaptic weight) governed by the so-called Synaptic Plasticity (SP) process can either enhance or inhibit the transmitted signals. SP is believed to play a crucial role in human brain learning and memory processes [27], [28] and Spike Timing Dependent Plasticity (STDP) is the fundamental process that enables transmission efficiency modulation based on the difference between the arrival times at the synapse locus of the pre-synaptic and post-synaptic spikes. Referring to Fig. 1 the synapse connecting neurons N_j and N_k, receives 2 input spikes S_j and S_k from the 2 neurons, and generates one output signal S_{jk}^{out} to be transmitted to neuron N_k. Its transmission efficiency W_{jk} is potentiated when the pre-synaptic spike S_j precedes the arrival of the post-synaptic spike S_k, and it is depressed when the incoming spikes arrival order is reversed. Long term plasticity (Long Term Potentiation - LTP and Long Term Depression - LTD) can manifest when such modulatory synaptic activity occurs over a longer period of time, inducing therefore a persistent history dependent influence on the synaptic transmission efficiency.

B. GRAPHENE-BASED SNN UNIT

The fundamental building block for our proposal in Section III is the graphene-based device depicted in Fig. 2. It relies on a Graphene NanoRibbon (GNR), which serves as a conduction channel when the device is subjected to a drain-to-source bias voltage $V_d - V_s$. The GNR device conduction profile is determined by the nanoribbon geometry and contact topology, while the actual conductance value is modulated by exerting external voltages on the top and/or back gates [29]. Fig. 2 depicts a GNR topology example and a conductance map example under variable back-gate and top gate voltages. The conduction map determines the device behavior and adequately tailored (e.g., topologically) GNR-based devices able to provide a rich set of functionalities for the implementation of Boolean gates [30] and neuronal/synaptic circuits [20], [22] have been reported. Furthermore, besides the conduction mechanism, it was experimentally observed that GNR-based devices exhibit intrinsic interface charge trapping/detrapping phenomena [31], i.e., depending on the applied gate voltage, charges are trapped/released at the graphene-oxide interface. As a result, the GNR conductance becomes dependent on device history via cumulated interface charges, which renders GNR-based devices appropriate for emulating neuron membrane potential dynamics and synapse plasticity functionalities [20], [22].

Fig. 3(a) schematically illustrates the graphene-based SNN unit [23], which implements a LIF neuron and a synapse with timing dependent plasticity via 4 pairs of GNR-based devices. To enable the basic SNN unit behavior, different nonlinear functionalities are required, and for every such functionality, a GNR device with a different shape is specifically devised, as discussed in [23].

The *synapse* core functionality is provided by Block 1, which receives input spikes from both the pre-synaptic (V_{spike}) and post synaptic (V_{feedback}) neurons, and generates the post-synaptic neuron input signal (V_{in}), with waveforms exemplified in Fig. 3(b). Initial synaptic weight values can be set through $\text{GNR}_{\text{up}}^{1}$ and $\text{GNR}_{\text{dn}}^{1}$ back-gate bias voltages, V_{up} and V_{dn}, respectively. As for synaptic plasticity, the $\text{GNR}_{\text{up}}^{1}$ cumulated trapped charges emulate long-term plasticity, while pair-wise STDP modulates the synapse output signal V_{in} amplitude based on the timing difference between the pre-spike V_{spike} and post-spike V_{feedback} occurrences, as illustrated in Fig. 3(b). Furthermore, by properly adjusting the back-gate bias voltage V_{up}, the same synapse can exhibit both Long-Term Potentiation (LTP) and Long-Term Depression (LTD), e.g., 100 mV for LTP and −100 mV for LTD. The GNR_{dn} back-gate voltage controls the inhibitory synaptic ability, i.e.,

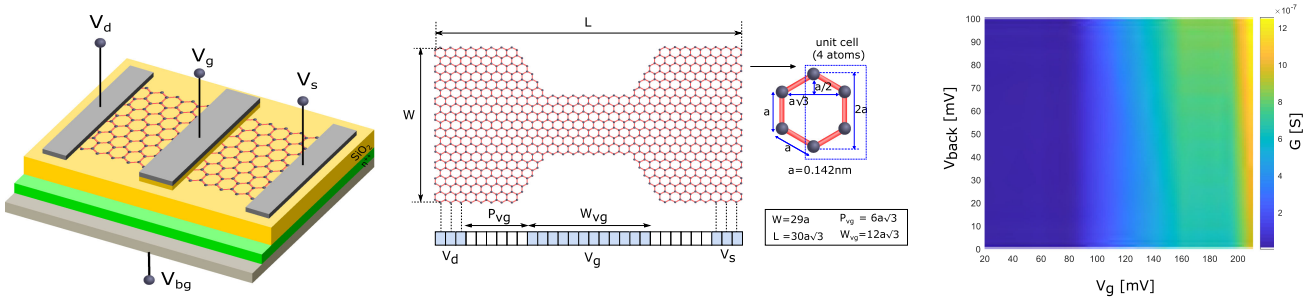
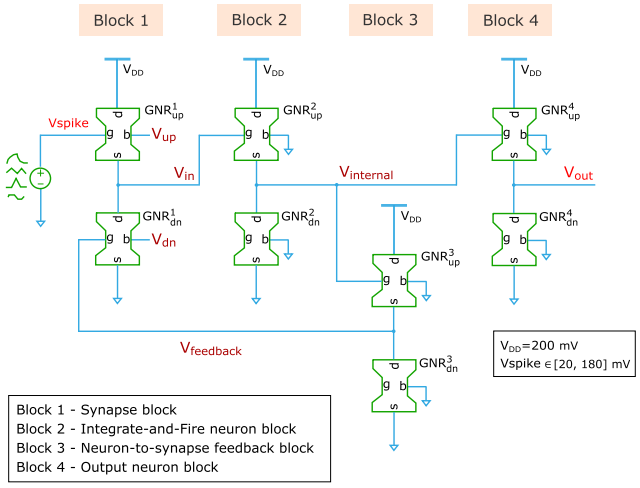
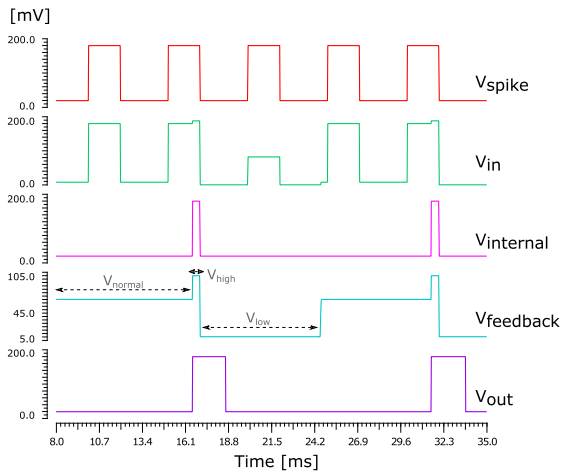


FIGURE 2. **Generic graphene-based device and conduction map example.**



(a)



(b)

FIGURE 3. **SNN unit: (a) circuit (b) basic operation.**

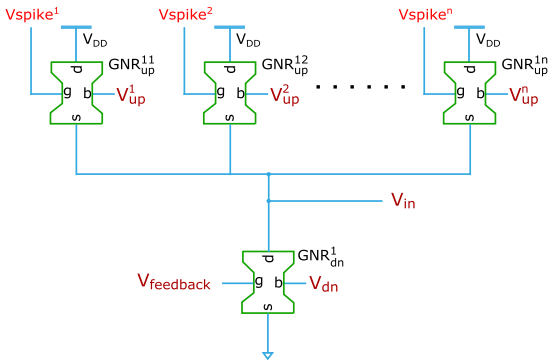


FIGURE 4. **Multiple input synapse block.**

0 mV for no inhibition and 180 mV for inhibiting all incoming spikes.

The *LIF neuron* comprises the remaining blocks 2, 3 and 4. Block 2 receives V_{in} as input signal from the synapse and is responsible for capturing the integrate-and-fire membrane potential dynamics. Charges trapped into GNR^2_{up} gate oxide can cause an equivalent voltage shift denoted as ΔV. When $V_{in} + \Delta V$ reaches the firing threshold, block 2 signals a firing event occurrence via the $V_{internal}$ signal. We note that block 2 suffices to emulate the spiking neuron's dynamics. However, since the neuron spike plays an important role for the STDP process, $V_{internal}$ is further processed by block 3 in order to send a post-spike back to the synapse to activate the pairwise STDP plasticity. Furthermore, to enable direct cascading of SNN units, $V_{internal}$ is also processed by block 4, which generates an output spike V_{out} that is compatible with the input spike V_{spike} in terms of voltage range (20 mV to 180 mV) and duration (2 ms). Note that the GNR topology details for all the SNN circuit devices are presented in [23].

As the basic SNN unit includes one synapse it can get input from one other neuron only while in practically relevant SNNs a neuron can be connected to hundreds of other neurons. To accommodate larger than one fan-in (i.e., n synapses connected to the same neuron) situations, the synapse block can be extended by replacing GNR^1_{up} with n GNRs in parallel as illustrated in Fig. 4 [23]. The synapses joint output voltage can then be derived as:

$$V_{in} = V_{DD} \cdot \frac{G^{11}_{up} + G^{12}_{up} + \cdots + G^{1n}_{up}}{G^{11}_{up} + G^{12}_{up} + \cdots + G^{1n}_{up} + G^{1}_{dn}}, \qquad (1)$$

where G^{1n}_{up} denotes the conductance of the n^{th} GNR_{up}.

C. SIMULATION FRAMEWORK

A hybrid framework combining atomistic-level simulation for graphene-based devices and circuit level SPICE simulation in Cadence [32] is utilized to properly evaluate the proposed reconfigurable graphene-based SNN architecture.

For the GNR device electronic transport properties calculation we make use of: (i) the Tight-Binding Hamiltonian to model the external potentials and the interactions between Carbon atoms, (ii) the Non-Equilibrium Green Function (NEGF) to solve the Schrödinger equation, and (iii) the Landauer-Büttiker formula to compute the graphene channel current and conductance [33]. The GNR potential distribution profile is obtained by solving a 3D Poisson equation self-consistently. Additionally, by calculating the equivalent voltage shift induced by interface trapped charges we account for the trapping/detrapping phenomena influence on the GNR device operation [34].

For the graphene-based SNN circuit evaluation, a Verilog-A GNR device simulation model [35] is employed. To enable high accuracy and time effective SPICE simulation, we make use of precomputed look-up tables containing atomistic level GNR simulation data for the utilized graphene-based devices. Additionally, we developed a Matlab simulation model to allow for the determination of the initial synaptic weight values according to the training method described in Section III-B. The obtained weights are subsequently converted into appropriate bias values that are utilized to initialize the SNN synaptic weights (via synapse GNR_{up} back-gate voltage) in the SPICE circuit model.

III. RECONFIGURABLE GRAPHENE-BASED SNN ARCHITECTURE

In this section we present the proposed reconfigurable graphene-based Spiking Neural Network (SNN) architecture, explain the mapping methodology of a generic SNN structure onto the proposed platform, and introduce a general training method for the determination of the initial synaptic weight values.

A. ARCHITECTURE OVERVIEW

Fig. 5 depicts a general overview of proposed reconfigurable graphene-based SNN platform. It mainly comprises a neuronal array, a synaptic array, and a peripheral Input/Output (I/O) block, which allows for SNN's communication with the computation platform (application) it embeds it. Specifically, the neuron array consists of N graphene-based spiking neurons, that can have their output connected either to the I/O block, when the neuron resides into the output SNN layer, or to the synapse array, when connecting with other layer neurons. The synapse array consists of $N \times N$ programmable graphene-based synapses, that can enable a connection either between neurons in different layers, or between the I/O block and SNN input layer neurons. The platform reconfiguration is enabled by means of: (i) a programmable switch matrix, which allows for SNN topology mapping onto the neuronal and synaptic arrays and (ii) an initialization module that programs the initial SNN network state (e.g., synaptic weights, plasticity types). The programmable switch matrix ensures that the signal routing within the neuronal and synaptic arrays is reflecting the desired SNN topology, by activating the appropriate interconnect crossbar row/column connections. The initialization module comprises a memory to store SNN's initial status data and a bias generator that decodes status information into voltage/current values to map the initial SNN status at electrical level. In particular, the memory module stores the synaptic weight initial values and plasticity type (e.g., LTP, LTD) for each SNN synapse while the bias generator converts these values into voltages to be applied to the GNR_{up} and GNR_{dn} back-gates of the corresponding physical synapse (as detailed in Section II-B).

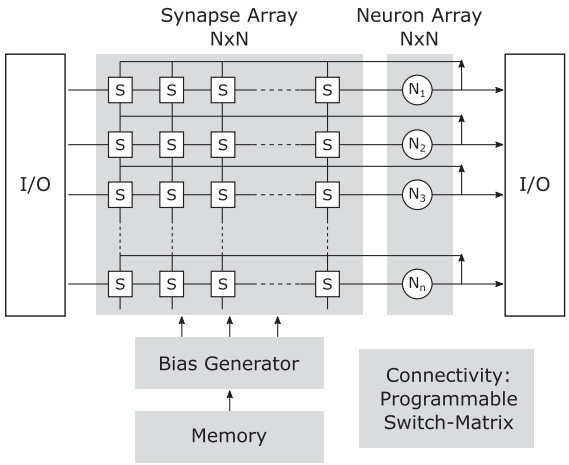


FIGURE 5. Reconfigurable graphene-based SNN platform.

B. ARCHITECTURE CONFIGURATION

To deploy a given application on the proposed reconfigurable graphene-based SNN architecture we make use of the following approach. First, we identify by means of state of the art approaches, e.g., [36], [37] an appropriate SNN topology for the considered application (e.g., number of layers, inter-layer connectivity, number of neurons per layer, synaptic plasticity types). Once the specific SNN topology is available, the SNN platform is reconfigured accordingly via the switch matrix. Subsequently, we identify an appropriate initial status of the SNN synaptic components by means of a training method able to determine suitable initial synaptic weight values. Finally, the per synapse weight value and plasticity type are transformed into bias voltages for the SNN synaptic array, and at this point, the SNN architecture is fully configured and ready to be utilized for the given application.

1) SNN Topology Mapping: To have a better inside on the mapping process, let us consider a 2-layer SNN consisting of 3 neurons in layer 1 and 1 output neuron in layer-2 as depicted Fig. 6. To structurally emulate this SNN on the proposed platform the neuronal array is configured such that neurons N_1, N_2, N_3 map the SNN layer 1 neurons and neuron N_4 maps the SNN layer 2 neuron. The left most column of synapses

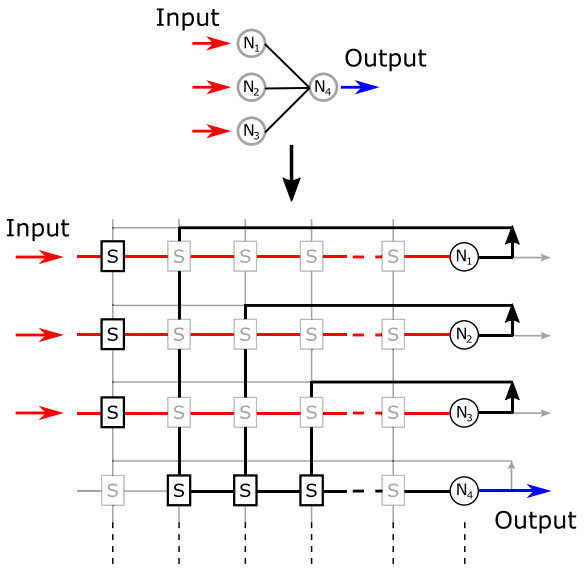


FIGURE 6. **SNN topology mapping example.**

are utilized for receiving SNN input and transmitting it to the layer 1 neurons. The layer 1 to layer 2 connectivity is enabled by the row of synapses corresponding to the N_4 neuron. N_4 is also connected further to the I/O block for SNN output readout.

2) SNN Initial Synaptic Weights Determination: For the identification of the initial synaptic weight values we propose the general training flow summarized in Algorithm 1, which assuming a given SNN topology and application specific input patterns, aims to identify the best set of initial synaptic weight values that can generate the desired SNN reaction for the applied inputs, e.g., for classification tasks, different inputs patterns should be discriminated by different output neurons. The training process can be divided into two stages: (i) *Stage* 1 (steps (1) to (4) in Algorithm 1), which concerns with the definition of the desired SNN reaction by labeling the output neurons according to the input patterns they react or should react to, and (ii) *Stage* 2 (steps (5) to (9) in Algorithm 1), during which the synaptic weight values are computed via an iterative process that minimizes the difference between the obtained and the desired SNN reaction. The training according to Algorithm 1 is carried out by means of Matlab.

In *Stage* 1, we define the desired SNN output response, e.g., which neuron should react to which input pattern such that all input patterns can be properly discriminated. To this end, we first instantiate the synaptic weights with random values (step (1)), apply the input patterns and obtain the SNN output response (step (2)). We then match all output neurons to the different SNN input patterns. Some neurons might already appropriately fire for the assumed input patterns and thus labeling them is straightforward, while others might not, case in which we enforce a label assignation. Once every output neuron has an assigned label (a designation for an SNN input category), the SNN desired reaction has been defined (step (4)).

In *Stage* 2, we update the synaptic weights repetitively until the SNN exhibits the desired output neuronal reaction for all input patterns. Specifically, the synaptic connections that contribute towards the desired SNN reaction are potentiated, while the connections that might trigger an undesired SNN reaction are depressed. For the sake of simplicity let us assume a 2 layer and 2 output SNN that has to classify input data according to two patterns P_1 and P_2. After *Stage* 1 the output neurons O_1 and O_2 are labeled as O_1 should react to P_1, and O_2 to P_2 while the current SNN reaction is that both O_1 and O_2 react to P_1, which is not the desired behaviour. To determine which synaptic weights should be potentiated and which ones should be depressed, we first determine the reaction of the layer-1 neurons. In particular, we identify the set of input neurons that are stimulated by P_1 and P_2, and denote them by G_1 and G_2, respectively. Since O_1 is already reacting only to P_1 as it should, we are interested in changing only the O_2 reaction via synaptic weights modification. To this end, we determine the difference set $G = G_2 - G_1$, which includes all input neurons that are excited by P_2 and not excited by P_1. Then, we: (i) potentiate the synaptic connections between the neurons belonging to the difference set G and the output neuron O_2 (since we desire O_2 to react for input pattern P_2), and (ii) depress all the synaptic connections between neurons belonging to G and output neuron O_1 (as O_1 shouldn't react for input pattern P_2). Fig. 7 illustrates an example for the synaptic weights updating process. To ensure the desired SNN reaction, i.e., O_1 reacts for P_1, and O_2 for P_2, (i) the synapses between the input neurons excited solely by P_2 and output neuron O_2 are potentiated (blue connections), and (ii) synapses from the input neurons excited solely by P_2 to output neuron O_1 are depressed (red connections).

We note that for larger SNN and problem size dimensionality, the weights update methodology described above can be applied in a sequential pairwise manner. The synaptic weights update is an iterative optimization process that ends when the difference between the current and the desired SNN output neuronal response is minimal. When completed, Stage 2 provides the set of initial synaptic weights values for the considered SNN topology.

Having generated the set of initial synaptic weights, the platform synapses GNR devices are biased, as described in Section II, with back gate voltages afferent to these weights, and at this point, the platform is fully configured.

IV. SIMULATION RESULTS

To demonstrate the suitability of proposed reconfigurable graphene-based SNN architecture for various application scenarios, as well as the plausibility of the synaptic weights training method, we consider two SNN topologies that are particularly designed for character recognition and edge detection, respectively, map them on the proposed reconfigurable graphene-based SNN architecture, and investigate their run-time performance by means of SPICE simulation. In both cases we make use of 2 ms input spike pulses varying from 20 mV to 180 mV and a supply voltage $V_{DD} = 200mV$. We

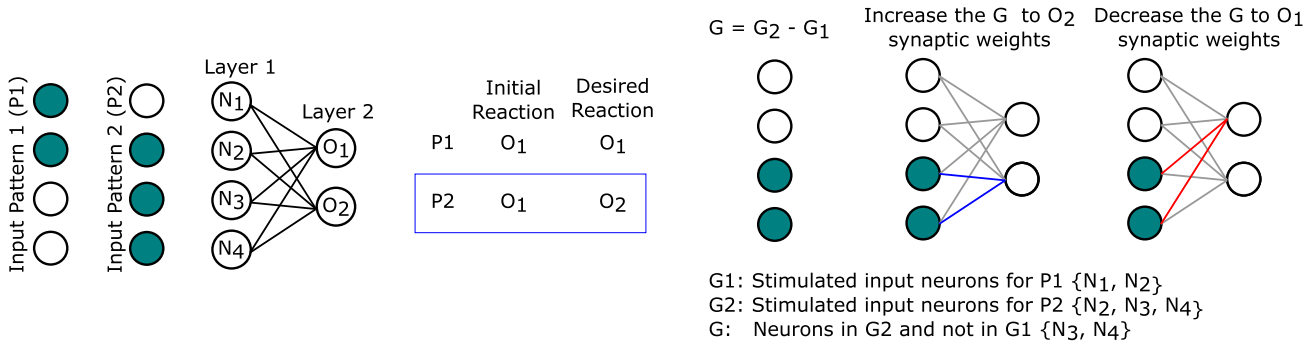


FIGURE 7. SNN synaptic weights training example.

Algorithm 1: Initial Synaptic Weight Values Determination.

Input: SNN topology & application specific input patterns
Output: Initial synaptic weight values

1: Randomly instantiate the synaptic weight values for the given SNN;
2: Apply input patterns to the SNN, and obtain the initial SNN output response;
3: Current SNN output neuronal response ← initial SNN output neuronal response from (2);
4: Determine the desired SNN response (output neurons labeling);
5: **while** current SNN output response ≠ SNN desired output response **do**
6: Update the synaptic weights based on the current and desired SNN output response (inhibit connections that might trigger undesired SNN reaction and enhance connections that help to produce the desired reaction);
7: Apply input patterns to the SNN with updated synaptic weights, obtain new SNN output response;
8: Current SNN output response ← new SNN output response obtained from (7).
9: **end while**

note however that our proposal is general and can be adapted to operate on different power supply values and input spike formats.

A. CHARACTER RECOGNITION

For character recognition we rely on the 2-layer SNN comprising 30 neurons depicted in Fig. 8, intended to recognize the vowel (and their variations) "A," "E," "I," "O," and "U". Each character is represented by a 5 × 5 black and white pixel matrix. The 25 neurons in layer 1 (L_1) serve as input neurons and each neuron corresponds to a pixel in the character matrix. For a given input character, each L_1 neuron receives input spikes if its corresponding pixel is black and no spikes if the

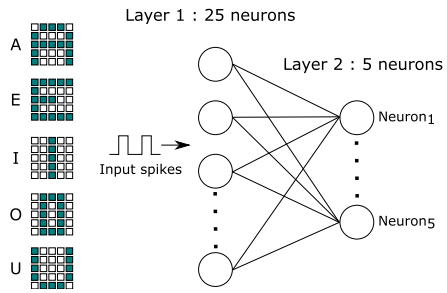
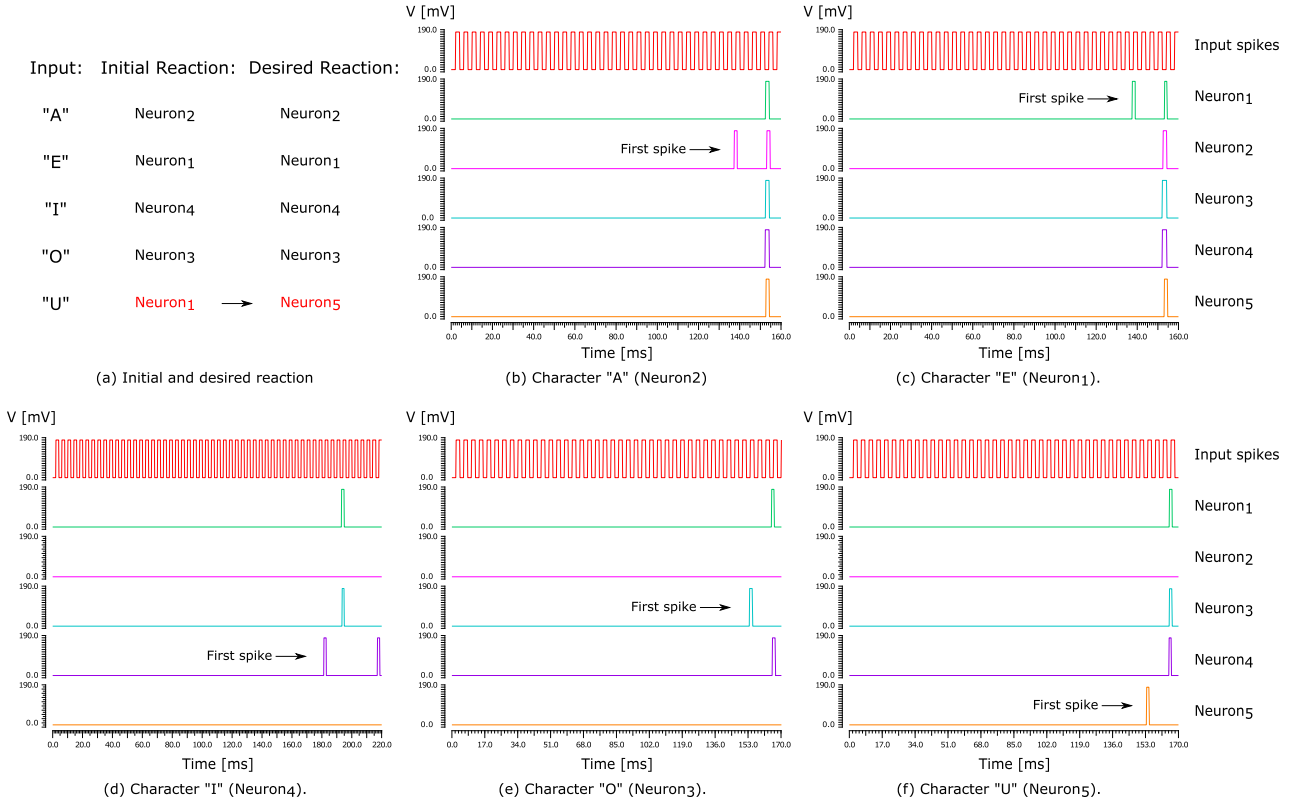


FIGURE 8. SNN for character recognition.

pixel is white. The 5 neurons in layer 2 (L_2) are output neurons, each one being meant to recognize a different character. In the considered SNN, we assume that the input pixels are fed to the L_1 neurons via LTP synapses with identical weights. As concerns the L_1 to L_2 connectivity every L_2 neuron is connected with all the L_1 neurons via LTP synapses with synaptic weight values determined by means of the proposed training method in Section III-B. When applying an input character, we stimulate the input neurons corresponding to the black pixels with identical 200Hz periodic input spike trains and employ "time-to-first-spike" scheme to indicate the recognition result, i.e., the output neuron that fires first is the one that recognized the input character.

To determine the weights of the L_1 to L_2 synapses, we first randomly instantiate them for every synapse. Then, we apply the 5 characters to the SNN one at a time and obtain the initial recognition results depicted in Fig. 9(a). The Figure indicates that "E" and "U" are both recognized by the same neuron Neuron_1, while each of the other 3 characters is recognized by a different unique output neuron. Based on this initial recognition results, we determine the desired SNN output neuronal response, i.e., characters "A," "E," "I," "O," and "U" should be recognized by neurons Neuron_2, Neuron_1, Neuron_4, Neuron_3, and Neuron_5, respectively. Thus, we now need to adjust synaptic weight values such that Neuron_5 recognizes "U" instead of Neuron_1 and preserve the SNN output reaction for the other 4 characters. To achieve this we update the synaptic weights of connections between input neurons stimulated by "U" and not by "E," and the output neuron



(a) Initial and desired reaction
(b) Character "A" (Neuron2)
(c) Character "E" (Neuron$_1$).
(d) Character "I" (Neuron$_4$).
(e) Character "O" (Neuron$_3$).
(f) Character "U" (Neuron$_5$).

FIGURE 9. **SNN recognition reaction for original vowel characters.**

Neuron$_1$ and/or Neuron$_5$ (depression and/or potentiation). After we obtain the desired SNN output neuronal response and so the initial values of the synaptic weights, we configure the graphene-based SNN architecture and set the initial state of the synaptic array. We do so, by adjusting the back-gate bias voltage (V_{up} in each synapse block, as detailed in Section II), which can take values between 0 mV and 200 mV with a 10 mV resolution. The SNN reaction for each character obtained by means of SPICE simulation of the configured graphene-based architecture is depicted in Fig. 9(b), (c), (d), (e), and (f), which clearly indicate that the obtained results are in line with desired recognition behaviour. As can be seen in Fig. 9(b)–(f), in all cases, initially, there are no firing events for the output neurons. After some time, one L_2 neuron fires (a different one for every character) and other output neurons may or may not fire afterwards. The reaction time for input characters "A," "E," "I," "O," and "U" are 135 ms, 135 ms, 180 ms, 150 ms, and 150 ms, respectively. As expected, the SNN exhibits longer reaction time for character "I" as it stimulates less input neurons in L_1 than the other characters.

The aforementioned simulation experiments utilized the 5 initial characters as input patterns. However, to get a more comprehensive assessment of the character recognition ability of the proposed graphene-based SNN, we extend the original 5 characters input patterns, with additional datasets containing variations of the original characters, obtained by adding 1, 2, or 3 extra pixels to the original characters, as exemplified in Fig. 10 for "A". Specifically, for each original character we

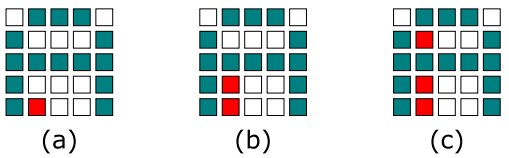


FIGURE 10. **"A" variations with additional pixels.**

TABLE I **Extended Dataset Cardinalities**

Variation Type	"A"	"E"	"I"	"O"	"U"
1 additional pixel	11	10	20	13	14
2 additional pixels	55	45	190	78	91
3 additional pixels	165	120	1140	286	364

generate datasets corresponding to each types of considered variation and Table I summarizes the dataset cardinality for each character and variation type. To investigate the effect of the initial synaptic weights values on the SNN recognition performance we derive 4 different initial synaptic configurations based on the following training sets: ($S1$) - the 5 original characters, ($S2$) - ($S1$) and 15 variations (1 new pattern per character for each variation type), ($S3$) - ($S1$) and 45 variations (2 new patterns per character for each variation type), and ($S4$) - ($S1$) and 75 variations (3 new patterns per character for each variation type). For each obtained initial synaptic configuration we instantiate the corresponding graphene-based SNN architecture and evaluate its recognition performance on a test

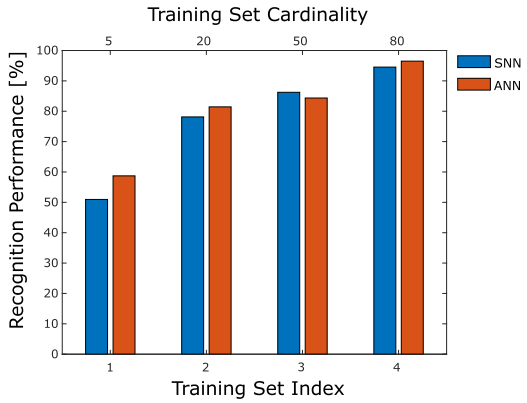


FIGURE 11. SNN vs. ANN character recognition performance.

dataset comprising all the character variations input patterns not employed in the corresponding training set.

Furthermore, to put our results into proper perspective we compare the obtained classification capabilities against the ones of an Artificial Neural Network (ANN), paradigm that is widely utilized for character recognition, trained (with the gradient descent method) and evaluated on the same datasets. The Matlab modelled ANN is a 3-layer feed forward network, with 25 input neurons, a hidden layer with 5 neurons, and an output layer with 5 neurons to indicate the recognized character. The ANN and graphene-based SNN character recognition performance is presented in Fig. 11. As can be observed in the Figure the recognition ability of both ANN and SNN improves for larger size training datasets, which is expected, from $\approx 55\%$ for training set $S1$ with cardinality 5, up to $\approx 95\%$ for training set $S4$ that contains 80 input patterns. Moreover, the SNN approach exhibits similar with ANN recognition performance (max. 7.8% variation), even outperforms ANN for the training set $S3$, while benefiting of all spike and graphene induced energy consumption and area advantages.

B. EDGE DETECTION

To further demonstrate the capabilities of the proposed reconfigurable SNN architecture we consider the 3-layer SNN comprising 13 neurons depicted in Fig. 12(a) and employ it to perform edge detection on the celebrated Lena and Cameraman images. To this end, for each and every image pixel, we should determine whether it belongs to an edge or not and this can be done by sequentially analyzing the pixel configuration of the 3×3 grayscale pixel matrix centered around it. Each layer 1 (L_1) neuron receives an input spike train which frequency is determined by the grey levels of the 3×3 matrix pixel it connects with. We assume that: (i) the input patterns are fed to the L_1 neurons via LTP synapses with identical synaptic weights, (ii) L_1 and L_2 neurons are fully connected via LTP synapses with initial weight values determined based on a set of directional filters that detect 3×3 edge and non-edge patterns, and (iii) for L_2 to L_3 connectivity, LTP synapses with identical synaptic weights are utilized to connect Neuron_1 and Neuron_2 to the output neuron, while an inhibitory connection is in place between Neuron_3 and the output neuron.

To determine the initial weight values for the synapses connecting L_1 and L_2, we first assign them random values. We use as SNN input patterns a series of edge and non-edge patterns [37] formalized as 3×3 grayscale pixels matrices, as depicted in Fig. 12(b). To represent the 3 grey levels (white, grey, black) in the considered edge and non-edge patterns, we use input spike trains with 0Hz, 190Hz, and 200Hz frequency, respectively. As desired SNN reaction, we would like the SNN output neuron to fire when an edge pattern is applied as input, and to not fire for non-edge input patterns. To induce the desired SNN output reaction, we need to update the L_1 to L_2 synaptic weights. As can be observed in Fig. 12(b), the edge patterns and non-edge patterns stimulate different numbers of input neurons. Specifically, the edge patterns stimulate 3 or 6 input neurons while the non-edge patterns stimulate 0, 1, 8 or 9 input neurons. Thus, we expect the non-edge patterns either to induce no firing event in L_2 neurons, or to induce firing events in more L_2 neurons than the edge patterns do. Therefore, we would like to take advantage of the L_2 neurons that are firing only for the non-edge patterns in order to induce the desired SNN output neuron reaction. Since this reaction for non-edge patterns is "do not fire," we can exploit the spiking of the neurons in L_2 that are firing only for non-edge patterns, in order to inhibit the SNN output neuron. In particular, we designate Neuron_3 in L_2 to fire only for non-edge patterns. Thus, we depress all incoming synaptic connections to Neuron_3 and inhibit its outgoing connection to the SNN output neuron. Since for this particular application, the initial synaptic weights values for desired SNN reaction to edge and non-edge patterns can be derived as previously described, we don't need to make use the methodology introduced in Section III. After obtaining the initial synaptic weights, the graphene-based SNN is configured accordingly, and the SNN reaction to the 18 3×3 edge and non-edge input patterns evaluated by means of SPICE simulation. The SNN reaction is summarized in Fig. 13(a) and, for exemplification purpose, graphically presented in Fig. 13(b)–(e), for edge pattern 1, edge pattern 5, non-edge pattern 2, and non-edge pattern 6, respectively. We note that for edge patterns at least one L_2 neuron is firing, which makes the SNN output neuron to fire while for the non-edge patterns, there are no firing events for the SNN output neuron, either because none of the L_2 neurons fire, or because the inhibitory neuron Neuron_3 fires and suppresses all the other L_2 to L_3 firing activity.

To evaluate the edge detection ability of the obtained SNN, we consider two grayscale images, i.e., Lena and Cameraman, depicted in Fig. 14(a) and (c), and rely on SPICE simulation to obtain the detection results. Prior to image processing they are first quantized in order to reduce the number of gray levels. Each pixel appurtenance or not to an edge is determined by the SNN processing of the 3×3 window centered in that pixel and after all pixels are scanned the edge detection result forms a black and white image, where black pixels belong to edges and white pixels don't.

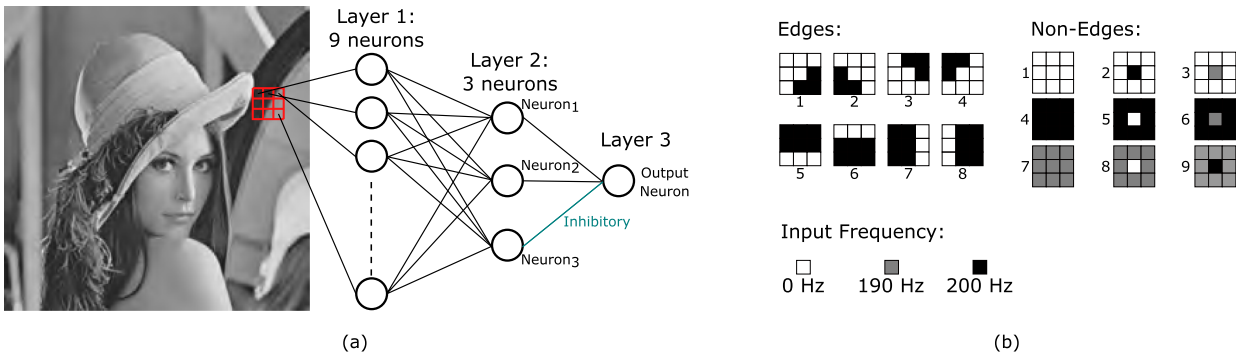


FIGURE 12. **Edge detection SNN illustration, (a) SNN structure, (b) edge patterns and non-edge patterns.**

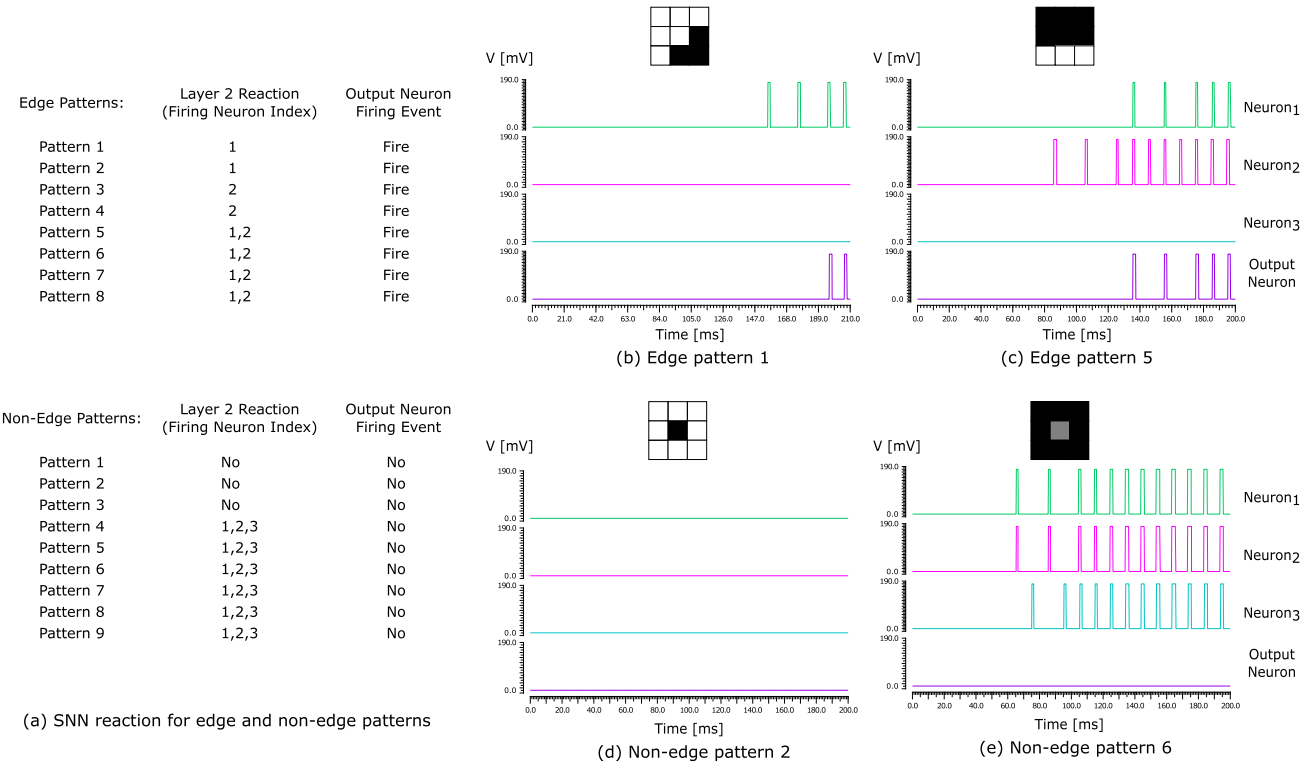
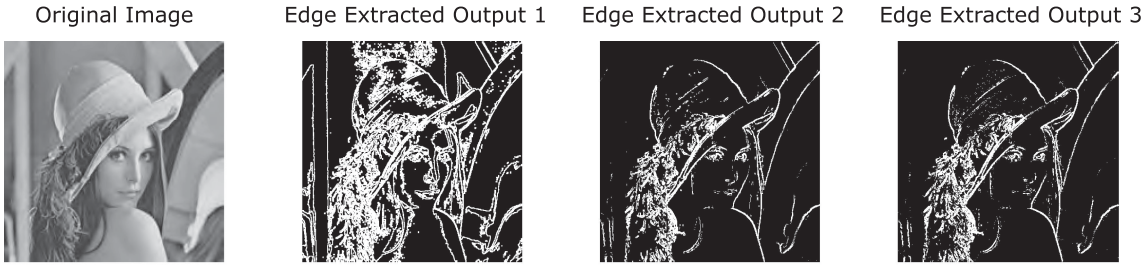


FIGURE 13. **SNN reaction for edge and non-edge input patterns.**

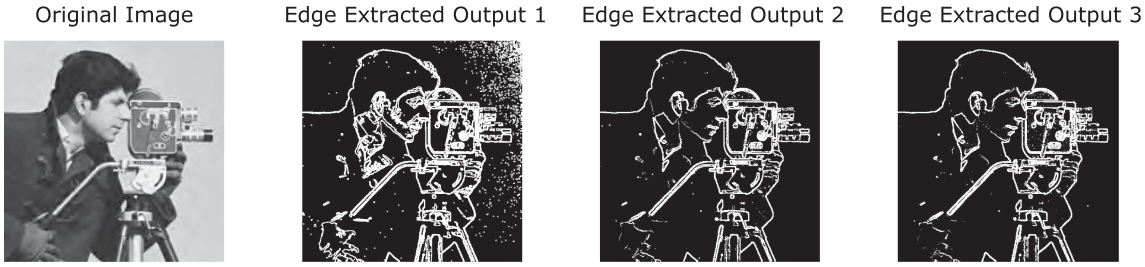
To get inside on the way quantization affects the edge detection performance, we perform edge detection on 4-, 8-, and 12-levels quantized images. We encode each grey level pixel as a spike train with a different frequency ranging from 0Hz (white) to 200Hz (black). Fig. 14(b) and (d) present the edge extraction results for different quantization levels for Lena and Cameraman, respectively. A visual inspection of Fig. 14 images reveals that 4 grey level quantization results in blurred edge images, while 8- and 12-level quantization in clear and sharp edges. Note that the image quality improvement becomes only marginal beyond a certain number of quantization levels (e.g., the difference between 8-level and 12-level quantized images generated edges are almost imperceptible).

To assess the quality of the SNN edge detection results we compare them against the Matlab obtained results for 4 classical edge detection operators, i.e., Canny, Roberts, Sobel, and Prewitt [38] applied directly on the original images (without prior quantization), in terms of the Peak Signal-to-Noise Ratio (PSNR) and the Mean Squared Error (MSE), which measure the perceptual distortion between the original images and the edge extracted counterparts [39]. We note that a higher PSNR indicates a higher quality image, while a lower MSE value promises a better image quality. Table II presents a comparative summary of the PSNR and MSE values computed for the edge extracted images, obtained with the classical edge detection algorithms and with the proposed graphene-based SNN when using different quantization levels. By inspecting the PSNR and MSE results for Lena image in Table II, we note that the best performing edge detector is Roberts and that the SNN counterpart exhibit only slightly worse performance:



(a) Lena image

(b) Edge extracted output 1, 2, and 3 for Lena image, which are obtained with 4, 8, and 12 quantization gray levels, respectively.



(c) Cameraman image

(d) Edge extracted output 1, 2, and 3 for Cameraman image, obtained with 4, 8, and 12 quantization gray levels, respectively.

FIGURE 14. **Edge detection results.**

TABLE II **Edge Detection Results Quantitative Evaluation**

	Lena Image		Cameraman Image	
	PSNR	MSE	PSNR	MSE
Canny	5.4486	$1.8545E+04$	4.6640	$2.2217E+04$
Roberts	5.5965	$1.7924E+04$	4.8678	$2.1198E+04$
Sobel	5.5718	$1.8026E+04$	4.8609	$2.1232E+04$
Prewitt	5.5708	$1.8030E+04$	4.8609	$2.1232E+04$
SNN 4 levels	5.1763	$1.9744E+04$	4.9972	$2.0576E+04$
SNN 8 levels	5.4381	$1.8589E+04$	4.8736	$2.1170E+04$
SNN 12 levels	5.4690	$1.8458E+04$	4.9798	$2.0658E+04$

(i) 7.5%, 2.8%, and 2.3% lower PSNR values and (ii) 10%, 3.7%, and 3% higher MSE values, for the SNN with 4, 8, and 12 gray levels, respectively. Looking at the SNN results, we note that in general the finer the gray level quantization the better the edge image quality, which is in agreement with the visual perception one gets when inspecting Fig. 14(b). For the Cameraman image, the SNN detector outperforms the classical counterparts, with Roberts fairing the best among the classical detectors. In particular, when comparing to Roberts figures, the SNN provides (i) 2.7%, 0.1%, and 2.3% higher PSNR values and (ii) 2.9%, 0.1%, and 2.6% lower MSE values for the SNN with 4, 8, and 12 gray levels, respectively. To conclude, the edge detection simulation results indicate that the graphene-based SNN platform delivers comparable performance with classical edge detectors for the considered Lena and Cameraman images, while providing all the benefits of SNN base processing paradigm.

C. AREA AND ENERGY EVALUATION

While an accurate evaluation of the area and energy consumption of our proposal is not possible at this stage of development it is of interest to get some inside on those two aspects. As it concerns the SNN unit area, a graphene-based synapse occupies an active area of ≈45nm ^{2} (2 GNR devices) and a neuron requires an active area of ≈176nm^2 (6 GNR devices) [23]. To evaluate the energy consumption and get sight into energy expenditures at different time scales, we considered an SNN unit and apply rectangular spikes with 40% duty cycle as input, while varying the input spike frequency within the range of 200Hz to 20GHz. We then measure the energy required by the SNN unit neuron to generate a spike and by the SNN unit synapse to perform plasticity modulation and spike transmission. The obtained results graphically presented in Fig. 15 indicate ≈8 orders of magnitude decrease in energy consumption per spike for both the neuron (from 43pJ at 200Hz to 5.2×10^{-7} pJ at 20GHz) and the synapse (from 5.1pJ at 200Hz to 6.0×10^{-8}pJ at 20GHz).

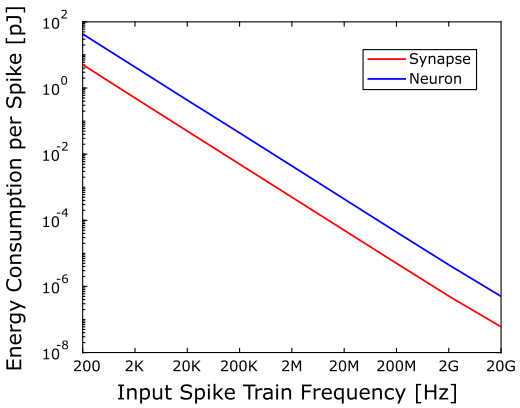


FIGURE 15. **SNN unit energy consumption vs. input spike train frequency.**

TABLE III Area and Energy Consumption for Biological and State-of-The-Art Artificial Neurons

	Neuron Type	Technology Type	Supply Voltage [V]	Area Spike [μm^2]	Energy per [pJ]	Spike Timescale
Biological	-	-	-	-	2.5×10^5	ms
2020 [40] biomimetic	Izhikevich	CMOS (65 nm)	0.3	1.5×10^2	2.4×10^{-2}	µs
2020 [40] processing	Izhikevich	CMOS (65 nm)	0.3	1.5×10^2	2.0×10^{-2}	µs
2017 [41]* biomimetic	ML	CMOS (65 nm)	0.2	2.0×10^2	7.8×10^{-2}	µs
2017 [41]* processing	ML	CMOS (65 nm)	0.2	3.5×10^1	4.0×10^{-3}	µs
2015 [42]	LIF	CMOS-SOI (20 nm)	0.8	9.0×10^{-2}	1.2×10^1	µs
2017 [43] biomimetic	LIF	CMOS-NEMS (28 nm)	0.5	8.0×10^{-2}	2.5×10^{-1}	µs
2018 [44]*	LIF	Memristor	-	-	1.0×10^{-2}	ns
Proposed	LIF	Graphene	0.2	1.8×10^{-4}	5.2×10^{-7} – 4.3×10^1	ps – ms

*Experimental results from fabricated devices.

TABLE IV Area and Energy Consumption for Biological and State-of-The-Art Artificial Synapses

	Synapse Type	Technology	Operating Voltage [V]	Area [μm^2]	Energy/Spike [pJ]	Spike Timescale
Biological	-	-	Spike: [−0.04,0.07]	-	1.0 – 1.0×10^1	ms
2020 [45]	STDP	CMOS (65 nm)	Supply: 1.2	-	1.0×10^{-3} – 5.0×10^{-2}	µs
2015 [46]*	STDP	CMOS (28 nm)	Supply: 0.2	1.3×10^2	2.3×10^{-6} – 3.0×10^{-5}	ms
2017 [47]*	STDP	Memristor	Spike: [−5,1.5]	1.0×10^{-2}	-	ms
2020 [48]	LTP	PCM	Control: 2.5	-	5.0 – 3.0×10^1	ns
Proposed	STDP+LTP	Graphene	Supply: 0.2 Spike: [0.02,0.18]	4.5×10^{-5}	6.0×10^{-8} – 5.1	ps – ms

*Experimental results from fabricated devices.

Further, to have a better view on the potential of using the proposed graphene-based SNN architecture for large-scale energy efficient implementations, we summarize in Table III and IV the area and energy consumption figures for biological and state-of-the-art artificial neurons and synapses, respectively. We note that, the graphene-based SNN unit (neuron+synapse) can potentially save at least ≈2 orders of magnitude area estate when compared with both neurons and synapses state-of-the-art implementations.

From the energy standpoint, the proposed SNN unit can consume up to 1 order of magnitude more than state-of-the-art counterparts if operated with spike pulse width in the order of, e.g., μs due to leakage but can achieve up to 4 orders of magnitude energy savings if operated with short spike pulse width in the order of, e.g., ps. For the presented applications, we considered a biologically plausible time scale, i.e., ms, and, as such, we obtained an average energy consumption for the entire SNN architecture neuronal and synaptic arrays of 1.98×10^4pJ per character for the vowels recognition application, and of 1.21×10^4pJ and 1.39×10^4pJ per edge pixel and non-edge pixel, respectively, for the edge detection application.

Note that the proposed SNN unit is generic thus it is by no means restricted to the considered design constraints. The SNN architecture can be tailored to function under different spike width scales by considering different trapping/de-trapping time constant values for the GNRs of the neuronal and synaptic array devices. Thus, we can target both biological, which require low input frequency and a specific voltage ranges, and fast processing scenarios for which a ns timing scale operation would be more appropriate from the computation speed point of view. To accommodate different application targets, the SNN architecture neuronal and synaptic arrays can be partitioned into different frequency islands, i.e., for each frequency island the neurons and synapses GNRs are designed with trapping mechanisms that match the island time scale. At run-time, depending on the application constraints, one can map the SNN topology either to the higher frequency islands for fast non-cortical processing, or to the lower frequency ones for bio-mimetic processing.

V. CONCLUSION

In this paper, we proposed a reconfigurable graphene-based Spiking Neural Network (SNN) architecture and a training methodology for obtaining the initial SNN synaptic weight values. The proposed architecture supports artificial synapses with programmable plasticity and reconfigurable connectivity between the neuronal and synaptic arrays. To investigate the versatility and suitability of the proposed architecture to accommodate and evaluate the behaviour of different SNN topologies we considered 2 SNN applications particularly designed for character recognition and edge detection. We mapped on the generic graphene-based platform a 2-layer SNN tailored for vowel characters recognition and demonstrated by mens of SPICE simulations that it can achieve up to 94.5% recognition accuracy for the considered training and evaluation datasets, which is very close to the one achieved by a functionally equivalent ANN counterpart. Further, we mapped and evaluated a 3-layer SNN to perform edge detection on Lena and Cameraman images and demonstrated

that the edge detection result quality matches and even outperform the one obtained with classical edge detection operators. In summary, the proposed reconfigurable graphene SNN architecture exhibits: (i) area and energy efficiency due to effective graphene-based implementation of neurons and synapses, (ii) flexible support for SNN applications mapping due to FPGA-alike reconfiguration feature, and (iii) training process simplicity due to Spike-Timing-Dependent Plasticity (STDP) and Long-Term Plasticity (LTP) support.

REFERENCES

[1] C. Mead, "Neuromorphic electronic systems," *Proc. IEEE*, vol. 78, no. 10, pp. 1629–1636, Oct. 1990.

[2] C. D. Schuman *et al.*, "A survey of neuromorphic computing and neural networks in hardware," 2017, *arXiv:1705.06963*.

[3] 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.

[4] H. Paugam-Moisy and S. M. Bohte, "Computing with spiking neuron networks," *Handbook Natural Comput.*, vol. 1, pp. 1–47, 2012.

[5] N. Qiao *et al.*, "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128 k synapses," *Front. Neurosci.*, vol. 9, no. 141, pp. 1–17, 2015.

[6] F. Akopyan *et al.*, "Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE Trans. Comput.-aided Des. Integr. Circuits Syst.*, vol. 34, no. 10, pp. 1537–1557, Oct. 2015.

[7] S. B. Furber *et al.*, "Overview of the spinnaker system architecture," *IEEE Trans. Comput.*, vol. 62, no. 12, pp. 2454–2467, Dec. 2013.

[8] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, no. 7191, pp. 80–83, 2008.

[9] H.-S. P. Wong *et al.*, "Phase change memory," *Proc. IEEE*, vol. 98, no. 12, pp. 2201–2227, Dec. 2010.

[10] Y. Babacan, F. Kaçar, and K. Gürkan, "A spiking and bursting neuron circuit based on memristor," *Neurocomputing*, vol. 203, pp. 86–91, 2016.

[11] B. Linares-Barranco and T. Serrano-Gotarredona, "Memristance can explain spike-time-dependent-plasticity in neural synapses," *Nature Prec.*, pp. 1–4, 2009.

[12] S. Ambrogio *et al.*, "Unsupervised learning by spike timing dependent plasticity in phase change memory (PCM) synapses," *Front. Neurosci.*, vol. 10, no. 56, pp. 1–12, 2016.

[13] K. D. Cantley, A. Subramaniam, H. J. Stiegler, R. A. Chapman, and E. M. Vogel, "Hebbian learning in spiking neural networks with nanocrystalline silicon tfts and memristive synapses," *IEEE Trans. Nanotechnol.*, vol. 10, no. 5, pp. 1066–1073, Sep. 2011.

[14] M. Chu *et al.*, "Neuromorphic hardware system for visual pattern recognition with memristor array and CMOS neuron," *IEEE Trans. Ind. Electron.*, vol. 62, no. 4, pp. 2410–2419, Apr. 2015.

[15] Y. Kim, Y. Zhang, and P. Li, "A reconfigurable digital neuromorphic processor with memristive synaptic crossbar for cognitive computing," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 11, no. 4, pp. 1–25, 2015.

[16] D. Chabi, Z. Wang, C. Bennett, J.-O. Klein, and W. Zhao, "Ultrahigh density memristor neural crossbar for on-chip supervised learning," *IEEE Trans. Nanotechnol.*, vol. 14, no. 6, pp. 954–962, Nov. 2015.

[17] N. Zheng and P. Mazumder, "Learning in memristor crossbar-based spiking neural networks through modulation of weight-dependent spike-timing-dependent plasticity," *IEEE Trans. Nanotechnol.*, vol. 17, no. 3, pp. 520–532, May 2018.

[18] P. Avouris and C. Dimitrakopoulos, "Graphene: Synthesis and applications," *Mater. Today*, vol. 15, no. 3, pp. 86–97, 2012.

[19] M. J. Allen, V. C. Tung, and R. B. Kaner, "Honeycomb carbon: A review of graphene," *Chem. Rev.*, vol. 110, no. 1, pp. 132–145, 2010.

[20] H. Wang, N. C. Laurenciu, Y. Jiang, and S. D. Cotofana, "Graphene nanoribbon-based synapses with versatile plasticity," in *Proc. IEEE/ACM Int. Symp. Nanoscale Architectures*, 2019, pp. 1–6.

[21] H. Tian *et al.*, "Graphene dynamic synapse with modulatable plasticity," *Nano Lett.*, vol. 15, no. 12, pp. 8013–8019, 2015.

[22] H. Wang, N. C. Laurenciu, Y. Jiang, and S. Cotofana, "Ultra-compact, entirely graphene-based nonlinear leaky integrate-and-fire spiking neuron," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2020, pp. 1–5.

[23] H. Wang, N. Cucu Laurenciu, Y. Jiang, and S. D. Cotofana, "Compact graphene-based spiking neural network with unsupervised learning capabilities," *IEEE Open J. Nanotechnol.*, vol. 1, pp. 135–144, 2020.

[24] R. C. Gonzalez and R. E. Woods, Digital Image Processing, 4th ed. London, U.K.: Pearson, 2018.

[25] E. Izhikevich, "Which model to use for cortical spiking neurons?," *IEEE Trans. Neural Netw.*, vol. 15, no. 5, pp. 1063–1070, Sep. 2004.

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

[27] L. F. Abbott and S. B. Nelson, "Synaptic plasticity: Taming the beast," *Nature Neurosci.*, vol. 3, no. 11, pp. 1178–1183, 2000.

[28] S. J. Martin, P. D. Grimwood, and R. G. Morris, "Synaptic plasticity and memory: An evaluation of the hypothesis," *Annu. Rev. Neurosci.*, vol. 23, no. 1, pp. 649–711, 2000.

[29] S. Cotofana, P. Dimitrakis, M. Enachescu, I. Karafyllidis, A. Rubio, and G. C. Sirakoulis, "On graphene nanoribbon-based nanoelectronic circuits viability," in *Proc. 42nd Workshop Compound Semicond. Devices Integr. Circuits Held Eur.*, 2018, pp. 35–36.

[30] Y. Jiang, N. Cucu Laurenciu, H. Wang, and S. D. Cotofana, "Graphene nanoribbon based complementary logic gates and circuits," *IEEE Trans. Nanotechnol.*, vol. 18, pp. 278–298, 2019.

[31] A. C. Ferrari *et al.*, "Science and technology roadmap for graphene, related two-dimensional crystals, and hybrid systems," *Nanoscale*, vol. 7, no. 11, pp. 4598–4810, 2015.

[32] Cadence. [Online]. Available: https://www.cadence.com/

[33] S. Datta, *Quantum Transport: Atom to Transistor*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[34] H. Wang, N. C. Laurenciu, Y. Jiang, and S. D. Cotofana, "Atomistic-level hysteresis-aware graphene structures electron transport model," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2019, pp. 1–5.

[35] Y. Jiang, N. C. Laurenciu, and S. Cotofana, "Non-equilibrium green function-based verilog-a graphene nanoribbon model," in *Proc. IEEE 18th Int. Conf. Nanotechnol.*, 2018, pp. 1–4.

[36] A. Gupta and L. N. Long, "Character recognition using spiking neural networks," in *Proc. Int. Joint Conf. Neural Netw.*, 2007, pp. 53–58.

[37] D. Lu, X.-H. Yu, X. Jin, B. Li, Q. Chen, and J. Zhu, "Neural network based edge detection for automated medical diagnosis," in *Proc. IEEE Int. Conf. Inf. Automat.*, 2011, pp. 343–348.

[38] G. Shrivakshan and C. Chandrasekar, "A comparison of various edge detection techniques used in image processing," *Int. J. Comput. Sci. Issues*, vol. 9, no. 5, pp. 269–276, 2012.

[39] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[40] S. Szczesny and D. Huderek, "60 pw 20um size CMOS implementation of an actual soma membrane," *J. Comput. Electron.*, vol. 19, no. 1, pp. 242–252, 2020.

[41] I. Sourikopoulos *et al.*, "A 4-fj/spike artificial neuron in 65 nm CMOS technology," *Front. Neurosci.*, vol. 11, no. 123, pp. 1–14, 2017.

[42] V. Ostwal, R. Meshram, B. Rajendran, and U. Ganguly, "An ultra-compact and low power neuron based on SOI platform," in *Proc. Int. Symp. VLSI Technol., Syst. Appl.*, 2015, pp. 1–2.

[43] S. Moradi, S. A. Bhave, and R. Manohar, "Energy-efficient hybrid CMOS-nems lif neuron circuit in 28 nm CMOS process," in *Proc. IEEE Symp. Ser. Comput. Intell.*, 2017, pp. 1–5.

[44] Y. Zhang *et al.*, "Highly compact artificial memristive neuron with low energy consumption," *Small*, vol. 14, no. 51, 2018, Art. no. 1802188.

[45] C. Huan, Y. Wang, X. Cui, and X. Zhang, "A reconfigurable mixed signal CMOS design for multiple stdp learning rules," in *Proc. IEEE 3rd Int. Conf. Electron. Technol.*, 2020, pp. 639–643.

[46] C. Mayr *et al.*, "A biological-realtime neuromorphic system in 28 nm CMOS using low-leakage switched capacitor circuits," *IEEE Trans. Biomed. Circuits Syst.*, vol. 10, no. 1, pp. 243–254, Feb. 2016.

[47] Z. Wang *et al.*, "Memristors with diffusive dynamics as synaptic emulators for neuromorphic computing," *Nature Mater.*, vol. 16, no. 1, pp. 101–108, 2017.

[48] D. Kaushik, U. Singh, U. Sahu, I. Sreedevi, and D. Bhowmik, "Comparing domain wall synapse with other non volatile memory devices for on-chip learning in analog hardware neural network," *AIP Adv.*, vol. 10, no. 2, p. 025111, 2020.