Progress in Artificial Intelligence (2024) 13:1–15
https://doi.org/10.1007/s13748-024-00313-4

REGULAR PAPER

A Deep Convolutional Spiking Neural Network for embedded applications

Amirhossein Javanshir[1] · Thanh Thi Nguyen[2] · M. A. Parvez Mahmud[3] · Abbas Z. Kouzani[1]

Received: 12 December 2022 / Accepted: 12 February 2024 / Published online: 13 March 2024

Abstract
Deep neural networks (DNNs) have received a great deal of interest in solving everyday tasks in recent years. However, their computational and energy costs limit their use on mobile and edge devices. The neuromorphic computing approach called spiking neural networks (SNNs) represents a potential solution for bridging the gap between performance and computational expense. Despite the potential benefits of energy efficiency, the current SNNs are being used with datasets such as MNIST, Fashion-MNIST, and CIFAR10, limiting their applications compared to DNNs. Therefore, the applicability of SNNs to real-world applications, such as scene classification and forecasting epileptic seizures, must be demonstrated yet. This paper develops a deep convolutional spiking neural network (DCSNN) for embedded applications. We explore a convolutional architecture, Visual Geometry Group (VGG16), to implement deeper SNNs. To train a spiking model, we convert the pre-trained VGG16 into corresponding spiking equivalents with nearly comparable performance to the original one. The trained weights of VGG16 were then transferred to the equivalent SNN architecture while performing a proper weight–threshold balancing. The model is evaluated in two case studies: land use and land cover classification, and epileptic seizure detection. Experimental results show a classification accuracy of 94.88%, and seizure detection specificity of 99.45% and a sensitivity of 95.06%. It is confirmed that conversion-based training SNNs are promising, and the benefits of DNNs, such as solving complex and real-world problems, become available to SNNs.

Keywords Spiking neural network · Satellite image classification · Epileptic seizure detection

1 Introduction

Following the rapid advancement of the mobile Internet and the Internet of Things (IoTs), the amount and complexity of the data are increasing exponentially. Meantime, driven by large amounts of data, technologies like artificial intelligence (AI) and cloud computing are growing [1]. A type of artificial neural network (ANN) model, named deep learning (DL), is gradually dominating the AI applications. Deep learning provides the state-of-the-art solutions for various applications, such as smart devices, IoT, autonomous vehicles, image classification, and object detection [2].

Despite the tremendous success of DNNs, a significant challenge of this technique is their requirement to a large amount of energy required when deployed on mobile and edge devices [3]. Accordingly, it is believed that a paradigm shift towards alternative algorithms and hardware solutions is necessary to their lower energy costs. Spiking neural networks (SNNs) try to mimic the brain's impulses (spikes) and local learning rules [4, 5]. They are proposed as the third generation of neural networks for embedded AI applications. SNNs are biologically inspired neurons that interact with each other via a series of spikes. ANNs, on the other hand, have continuous values rather than biologically plausible spikes [6]. Another distinction is the activation function. SNNs have discrete values and non-differentiable functions. Therefore, considering the event-driven computational and resource-consuming benefits of the spiking neuron model, SNNs are promising for low-power embedded applications

✉ Abbas Z. Kouzani
kouzani@deakin.edu.au

1 School of Engineering, Deakin University, Geelong, VIC 3216, Australia

2 Faculty of Information Technology, Monash University, Melbourne, VIC 3800, Australia

3 School of Mathematical and Physical Sciences, University of Technology, Sydney, NSW 2007, Australia

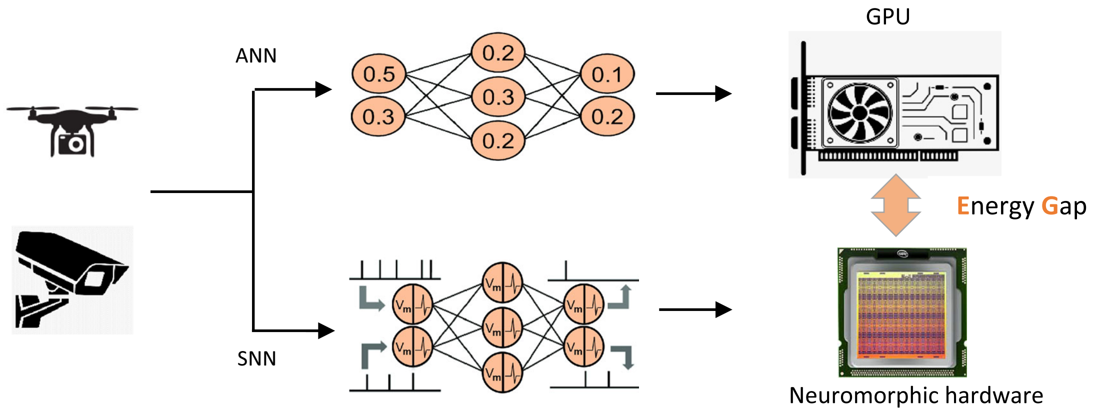


Fig. 1 Difference between an ANN (top) and a SNN (bottom) in terms of computation unit and hardware implementation

compared to conventional ANNs [7]. Figure 1 shows the differences between ANNs and SNNs.

In principle, SNNs can be used for the same applications as ANNs. Despite the potential advantages with respect to energy efficiency, because of the lower scalability of SNNs models, the research in the SNNs field has been typically focused on datasets, such as MNIST [8–11], N-MNIST [12–16], Fashion-MNIST [17–20], and CIFAR10 [21–25]. Therefore, this paper examines the applicability of SNNs to real-world applications, such as scene classification problems (satellite dataset) and forecasting epileptic seizure in the healthcare field.

Despite the outstanding performance of DNNs in predicting epileptic seizures [26–28] and satellite dataset classification [29–31], their implementation on smart devices (mobile medical devices) and on-board computers (CubeSat) are still limiting [32, 33]. This is due to the high energy consumption of DNNs.

In this work, we investigate the use of deep convolutional SNNs for two case studies: scene classification problems and epileptic seizure detection. The contributions made in this work are as follow:

- We develop a deep convolutional spiking neural network (DCSNN) to real-word classification problems.
- We evaluate the performance of DCSNN in two case studies: LULC classification (EuroSAT dataset) and epileptic seizure detection (CHB-MIT dataset).
- We evaluate the computational complexity and energy consumption of our DCSNN and against DCNN.

The rest of the paper is structured as follows. Section 2 describes the background on the fundamental components of deep convolutional SNNs. Section 3 describes the method of conversion of a deep convolutional neural networks (CNN) to a SNN. Section 3 describes theory of CNN to SNN conversion. Sections 4 and 5 describe the implementation of the first and second case study, respectively. Section 6 presents experimental results and proposes future improvements while Sect. 7 concludes this paper.

2 Fundamental components of deep convolutional SNNs

2.1 Convolutional Neural Networks

Convolutional neural network (CNN) is a special sort of ANNs that have fewer connections and parameters making their training easier while maintaining competent performances [34]. Figure 2 shows a classic CNN, LeNet, which includes four types of layers: an input layer, convolutional layers, pooling layers, and fully connected layers [35]. The capacity of the network can be controlled by varying its parameters such as the shape of the input (batch size, height, width, channels), the number of channels in the first and second convolution layers, the number of neurons in the hidden layer, and the kernel size for the convolution and pooling filters.

2.2 Information coding

The initial phase of an SNN is to transform the original features into spikes to feed the spiking networks. The encoding strategy has a significant effect on network performance. Selecting the best coding strategy depends on the application, hardware limitations, and the neuron model [36]. In general, three encoding mechanisms have been popularised with respect to input data.

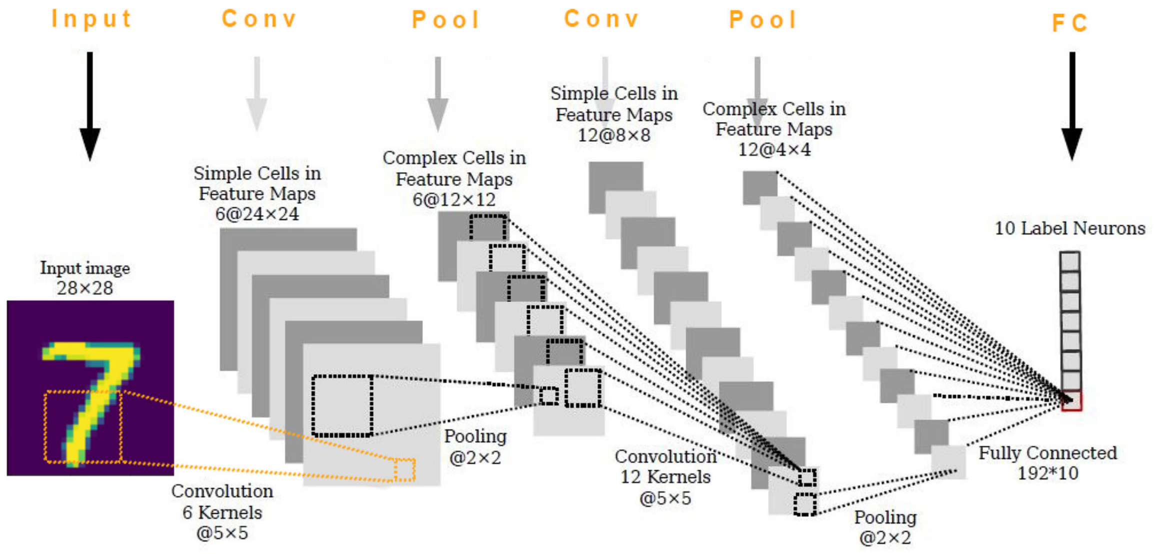

Input
Conv
Pool
Conv
Pool
FC
Input image 28×28
Simple Cells in Feature Maps 6@24×24
Complex Cells in Feature Maps 6@12×12
Simple Cells in Feature Maps 12@8×8
Complex Cells in Feature Maps 12@4×4
10 Label Neurons
Convolution 6 Kernels @5×5
Pooling @2×2
Convolution 12 Kernels @5×5
Pooling @2×2
Fully Connected 192*10

Fig. 2 A typical CNN architecture, consisting convolutional, pooling, and fully connected layers

1. Rate coding: this coding scheme comes from the observation of the relationship between stimulus intensity and neuronal firing rate. Rate-based coding converts input intensity to firing rate or spike count [37].
2. Population coding: in this coding scheme, the information is contained in the set of neurons that are active in response to a particular stimulus [38].
3. Temporal coding: this coding scheme cares about the individual timings of spikes, in opposition to rate-based coding. Information is encoded in the precise timing of spikes [39].

2.3 Neuron model for ANN

An ANN consists of at least two fundamental components: processing units (neurons) and connections (weights). Each neuron receives an input, uses its transfer function to process it, and then produces an output according to its activation function, as shown in Fig. 3a. The transfer function TF_J is the weighted sum of the inputs $I_{1\rightarrow M}$ with respect to the weights $\omega_{1\rightarrow M,J}$ of the links connected to the neuron J. Accordingly [40]:

$$\mathrm{net}_J = \mathrm{TF}_J(I, \omega) = \sum_{i=1}^{M} I_i.\omega_{i,J} \tag{1}$$

The output y_J of the Jth neuron is expressed as $y_J = \mathrm{AF}_J(\mathrm{net}_J)$ where the activation function AF_J is often the rectified linear units (ReLU) function.

2.4 Neuron model for SNN

The spiking neuron receives inputs from presynaptic neurons over multiple timesteps, while in ANNs, each neuron's state is updated periodically [41]. The leaky integrate-and-fire (LIF) is a first-order model and one of the simplest and most used neuron models, thanks to its computational efficiency. The basic concept of the LIF model is that the neuron has a membrane potential that evolves through time due to incoming excitatory or inhibitory currents and a leakage parameter [42]. An output spike occurs whenever the membrane potential of a neuron crosses its threshold, and its membrane potential is then reset to its resting state, as shown in Fig. 3b. This behaviour is characterised as:

$$V(t) = V(t-1) + L + X(t) \tag{2}$$

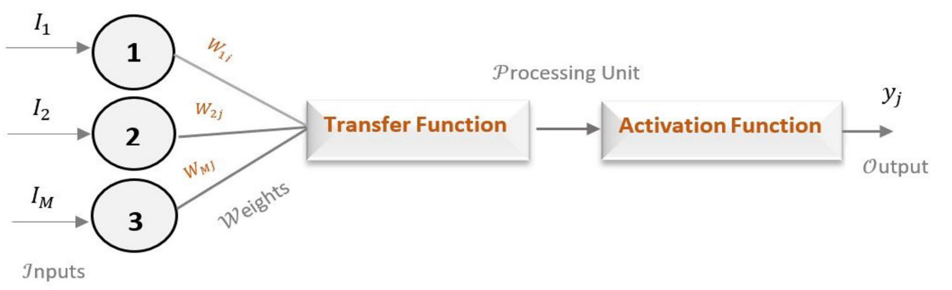
$$\text{If} \quad V(t) \geq \theta_{\text{threshold}}, \quad \text{Spike and reset } V(t) = V_{\text{rest}} \tag{3}$$

$$\text{If} \quad V(t) < V_{\text{rest}}, \quad \text{rest } V(t) = V_{\text{rest}} \tag{4}$$

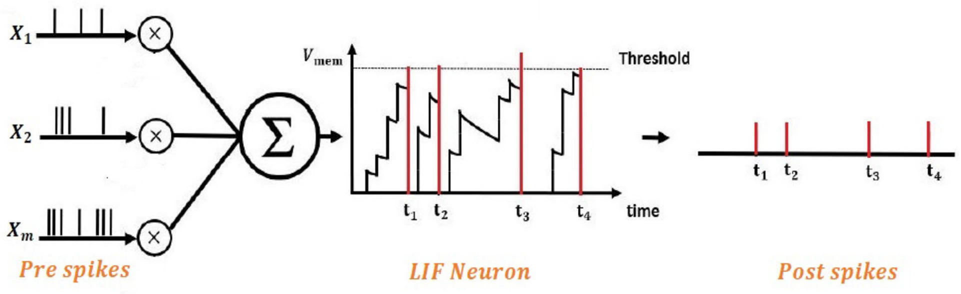
where $V(t)$ is the membrane potential, L is the leakage parameter, $X(t)$ is the summation of all synaptic inputs at time t, V_{rest} is the resting membrane potential, and $V_{threshold}$ is the membrane threshold. To propose a hardware friendly method, we use no leakage ($L = 0$), which gives the integrate-and-fire (IF) neuron model.

2.5 Modified leaky integrate-and-fire model

In this paper, we investigate a rate encoded SNNs, so the basic LIF model as presented in the previous section has an issue



a



b

Fig. 3 An overview of ANN and SNN operation, **a** inputs, outputs, and processing unit of ReLU neuron **b** inputs, outputs, procedure of integrating, fire, and reset of LIF neuron

when used for rate-based computations that happen close to zero: the output result is dependent on the order of arriving spikes, which is not desirable in a rate-based context. Specific care should be given to low firing rates and low threshold values. To illustrate this, let us consider a LIF neuron J with two different inputs I_i weighted by their respective weight w_i. In addition, assume that the $V_{resting}$ is set to zero. Let us suppose $w_1 = a < 0$ and $w_2 = b > 0$. If I_1 spikes before I_2, the output is $y_J = b$ as $y_J = (0-a)+b = 0+b = b$. However, if the order is reversed, $y_J = b-a$. This simple example highlights the limitations of this model. We applied an adapted model to fix this issue: setting the lower bound threshold $\theta^{down}_{threshold}$ to a negative value, $\theta^{\mathrm{down}}_{\mathrm{threshold}} = -\theta^{\mathrm{up}}_{\mathrm{threshold}}$ while keeping $v_{resting} = 0$ and $\theta^{\mathrm{up}}_{\mathrm{threshold}} = \theta_{\mathrm{threshold}}$. This modification mathematically solves the problem around zero. As a matter of fact, $y_J = b - a$ in both cases now. Yet, the problem still exists around $\theta^{\mathrm{down}}_{\mathrm{threshold}}$. V_m could be reset to v_{resting} without firing or simply kept at $\theta^{\mathrm{down}}_{\mathrm{threshold}}$ (selected in this work).

Once the spike is generated, the membrane potential is reset. We discuss next two types of resets: reset to zero presented by Diehl et al. [43], always sets the membrane potential back to a baseline, typically zero. Reset by subtraction, or "linear reset mode" presented by Diehl et al. (2016); Cassidy et al. (2013), subtracts the threshold V_{thr} from the membrane potential at the time when it exceeds the threshold [44, 45]. Accordingly, the adapted model inspired by similar considerations of work in [46–48] is given in Eqs. (5–8).

$$\mathrm{V(t) = V(t-1) + L + X(t)} \tag{5}$$

$$\text{If } \mathrm{V(t)} \geq \theta_{\mathrm{threshold}}, \text{ Spike and reset by subtraction } V(t) = V(t-1) - \theta^{\mathrm{up}}_{\mathrm{threshold}} \tag{6}$$

$$\text{If } \mathrm{V(t)} < \theta^{\mathrm{down}}_{\mathrm{threshold}}, \quad \text{rest } \mathrm{V(t)} = \theta^{\mathrm{down}}_{\mathrm{threshold}} \tag{7}$$

$$\theta^{\mathrm{down}}_{\mathrm{threshold}} = -\theta^{\mathrm{up}}_{\mathrm{threshold}}, \quad v_{\mathrm{resting}} = 0 \tag{8}$$

If $\theta^{\mathrm{down}}_{\mathrm{threshold}} = 0$ and $\theta^{\mathrm{up}}_{\mathrm{threshold}} = \theta_{\mathrm{threshold}}$, this model is equivalent to the basic LIF neuron model. In this work, the modified LIF model is used without leak, i.e. $L = 0$. We choose $F_{\min} = 50$ Hz and $F_{\max} = 200$ Hz in this work.

2.6 Rate-based coding

To convert the original features into spikes to feed the spiking networks, rate strategies are used for encoding information in SNNs, which is based on spike firing rate. To simulate neuron behaviours, a Poisson process is often used to generate spike trains. SNNs receive a series of spikes as input and produce a series of spikes as the output, which is usually referred to as spike trains. It is a process where events are assumed to be statistically independent which gives a good approximation of stochastic neuronal firing. The Poisson process is often used to convert the pixel illumination into a spike train: it

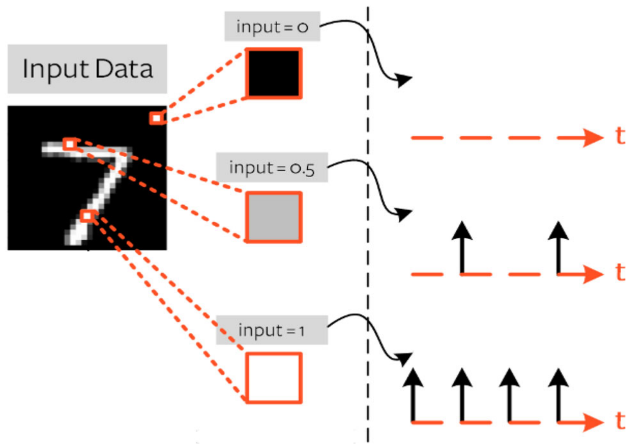


Fig. 4 Rate coded input pixel and spiking correspond to the pixel value. A black pixel will never produce a spike, while a white pixel corresponds to 100% of the spike

produces on average a spiking rate proportional to the illumination of pixels while introducing noise [49]. Figure 4 shows an example of the rate coding.

3 Theory of CNN to SNN conversion

The idea behind the conversion approach is to use the traditional training algorithms on a continuous-valued architecture such that optimal weights can be obtained and then transferred to an equivalent SNNs architecture. The key distinctions among ANNs and SNNs are the type of input and activation unit. ANN's inputs and activations are analogue values with no concept of time, whereas SNNs work with binary spike trains over time. The basic idea behind converting ANNs into SNNs is to match the firing rates of SNN neurons with the activation of the original ANN neurons [43, 50]. To understand the conversion method, we begin with relation between firing rates of spiking neurons and the activations of rectified linear units (ReLUs) in network conversion. Firstly, the ReLU can be considered a firing rate approximation of an IF neuron with no refractory period, whereby the output of the ReLU is proportional to the number of spikes produced by an IF neuron within a given time window. ReLU neurons computationally cheaper and they bear functional equivalence to an IF spiking neuron without any leak and refractory period [51]. This explains why this activation function is used to simulate an IF neuron, instead of the Tanh or the sigmoid functions, for instance. The inputs of an ANN are encoded as Poisson spike trains into the first hidden layer of a SNN. The approximation errors between the activation of ANN neurons and the firing rates of SNN neurons are the key source of performance loss during the conversion. CNN-to-SNN mapping has the following steps. Figure 5 shows the CNN-to-SNN mapping steps.

- Step 1: Modifying CNNs
- Step 2: Training CNN using Tensorflow-Keras
- Step 3: Transferring weights from the modified CNN to the equivalent SNN architecture while performing a proper weight–threshold balancing.

The first step of the mapping approach is to convert the original CNN into a modified CNN. Some modifications are required to make the static network convertible to spiking neurons:

- Bias neural units

Typically, bias neural units are used for ANNs, which allows the classifier to translate its decision boundary by a fixed value. A bias-less neural units constraint is included in our paper. This is due to the fact that including bias in a SNN, leads to accuracy loss during the conversion procedure, expanding the parameter space exploration and affecting the threshold voltage. By including a zero bias in the training process of the CNN, we ensure that SNN neurons are only defined by their threshold function and the synaptic weights, which reduce the complexity of the design space.

- Dropout

We used dropout to assist the regulation process during training. This technique eliminates a specific number of inputs to avoid overfitting.

- Pooling operation

The pooling operations are commonly employed in CNNs to reduce the dimensionality convolution map. Average pooling and max pooling are two common choices for performing the pooling operation. Max pooling is a commonly used function in CNN to spatially down-sample feature maps. However, because SNNs use binary activations rather than analogue values, the max pooling cannot be implemented. This is due to the fact that using max pooling could result in a considerable loss of information for the next layer. So, we use the average pooling in our design.

- SoftMax

The SoftMax function is used in CNN outputs. It generates a probability distribution of the output belonging to a particular class. In terms of SNNs, there are two ways to predict the output class. Firstly, predicting the output class associated with the neuron that spiked most. This method, however, is inefficient if all neurons in the last layer receive negative inputs. The second way is to replicate the behaviour of SoftMax in a spiking neuron. We used an external spike generator

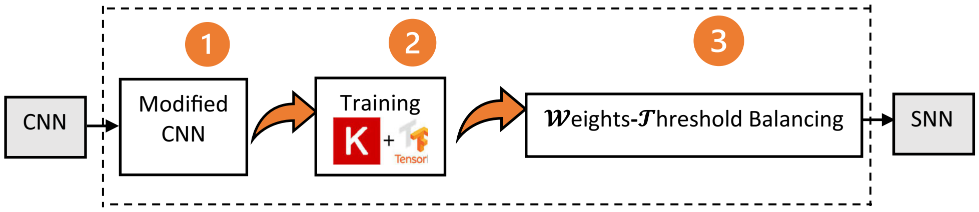


Fig. 5 Framework for CNN-to-SNN mapping

to generate spikes based on the weighted sum accumulated by each spiking neuron.

The second step is to train the modified CNN using the backpropagation method. The third and last step maps the weights from the ReLU network to a network of IF units and applies weight–threshold balancing to achieve faster convergence and less loss accuracy. The main challenge during the mapping approach is to avoid a substantial drop in classification accuracy after conversion. This loss comes from the following factors:

1. The input firing rates are too low such that the IF units do not cross their threshold, leading to a lower effective output.
2. The input firing rate is so high that the IF units output too many spikes

Therefore, an appropriate balancing between the firing threshold and the input weights is essential to ensure an accurate translation of continuous-valued neuron activations into firing rates of spiking neurons that makes the quality of the mapping greatly dependent on the ratio between the neuron threshold and the synaptic weights. The two common ways to achieve a proper rescaling are to use either a weight normalisation (WN) or threshold balancing (TB) proposed by Diehl et al. [43] and Sengupta et al. [52]. The former is used to regulate the spiking rate to decrease accuracy loss, and the latter is used to assign an appropriate threshold to the spiking neurons. The firing rates of the spiking neurons are limited to the range $(0, r_{\max})$ in SNNs simulation. On the other hand, ReLU activation has no such restrictions. To ensure that the firing rates are approximated by the activations, a weight normalisation approach is employed to normalise the ReLU activations from $(0, a^l_{\max})$ to $(0, r_{\max})$ by rescaling the convolutional layers weights. The weight normalisation can be represented as follows:

$$\overline{W^{\iota}} = \frac{\lambda^{\iota-1}}{\lambda^{\iota}} W^{\iota} \tag{9}$$

where λ^l is the max activation of layer l.

4 First case study: land use and land cover classification

Remote sensing (RS) technology has made rapid progress over the last decade. An unprecedented volume of data is available, and access to information is more straightforward. All of this gives us an understanding of the constant changes in the earth's surface, as well as the socio-ecological interactions that accompany them. Land use and land cover (LULC) data is critical for a wide range of geospatial applications, including city planning, local administration, and environmental management. Significant efforts have been made to study the possibility of applying AI, especially DNNs, on Earth Observation (EO) satellites. However, the limited energy budget of DNNs on hardware is one of the problems restricting the application of DNN models. This fact has motivated the study of energy efficiency networks such as SNNs to solve remote sensing tasks.

4.1 Methods

In this research, we initiated the experiments with one of the standard deep learning models—Visual Geometry Group (VGG16) network [53]. Figure 6 shows the conversion of the VGG16-based ANN convolutional block to an SNN convolutional block. Some changes are needed to be able to transfer the VGG16 network to spiking neurons. We eliminate the bias and batch normalisation, as they lead to accuracy loss during the conversion process. We also swap the max pooling with average pooling in our design. The second step is to train the VGG16 network using the backpropagation method in order to have a robust and accurate trained model. We apply data augmentation techniques including resizing, rotation, horizontal flip, and vertical flip. To reduce overfitting and enhance the generalisation capability of our model, firstly, we use dropout to assist the regularisation process during training due to the absence of bias and batch normalisation. Secondly, we apply early stopping as a regularisation technique to stop the training after a number of epochs when the limited validation dataset no longer improves the performance of the model. Thus, the best weights are saved and updated during training. The final step is transferring the trained weights to a network of IF units. A weight–threshold balancing technique is used to ensure an accurate translation of continuous-valued neuron activations into the firing

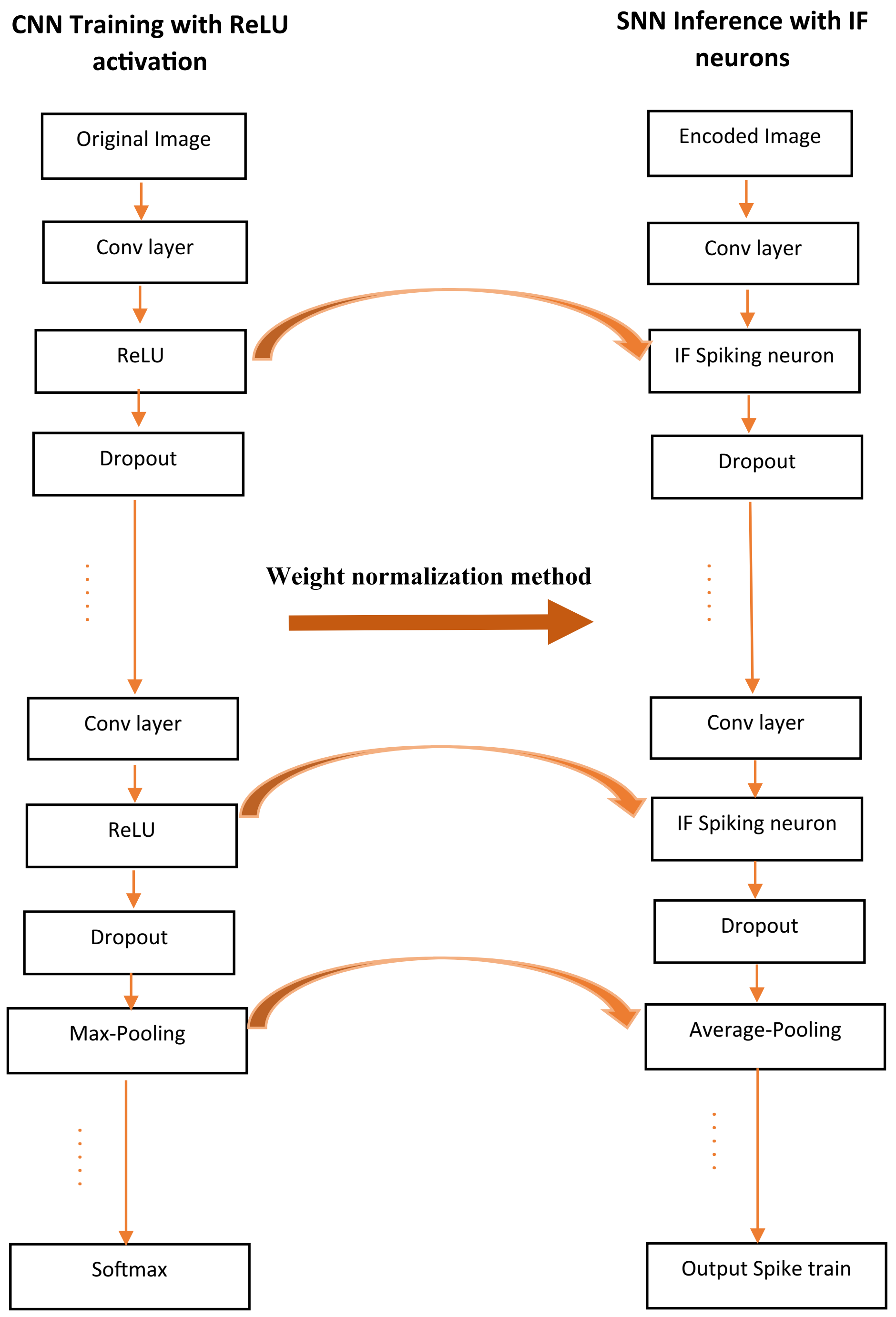


Fig. 6 Flowchart showing the conversion of the VGG16-based ANN convolutional block to an SNN convolutional block. From left to right. First, the max pooling layers in the original network are replaced with average pooling, and the network is trained. Dropout technique is used as a regularizer for both ANN and SNN. Then, the ReLU activations are replaced with IF spiking neuron

rates of a spiking neuron. To copy the behaviour of SoftMax to a spiking neuron, we apply an external spike generator. Figure 7 shows the conversion process of Deep CNN to Deep CSNN.

4.2 Dataset

The EuroSAT is a dataset for LULC classification taken from the Sentinel-2 satellite [54]. It includes 10 different scene classes with 27,000 labelled images, and 13 spectral bands. Examples of EuroSAT dataset are shown in Fig. 8. The dataset is divided into 80/10/10 ratios for training, validation, and test, respectively. We apply several augment techniques to avoid overfitting during the training. These techniques include random zoom, resizing, rotation, horizontal flip, and vertical flip.

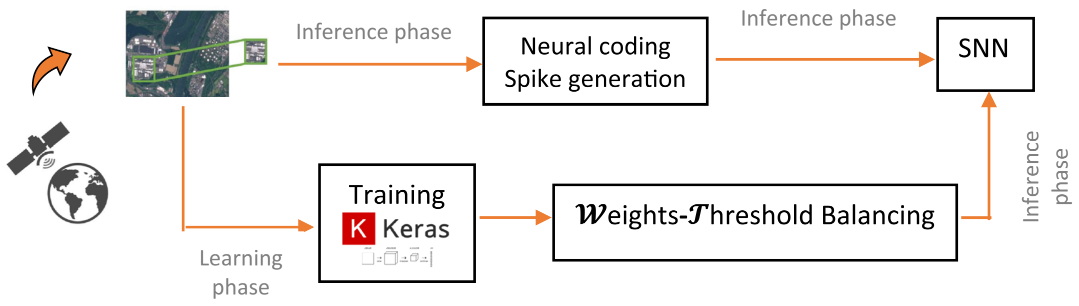


Fig. 7 Conversion process of the Deep CNN to Deep CSNN for LULC classification

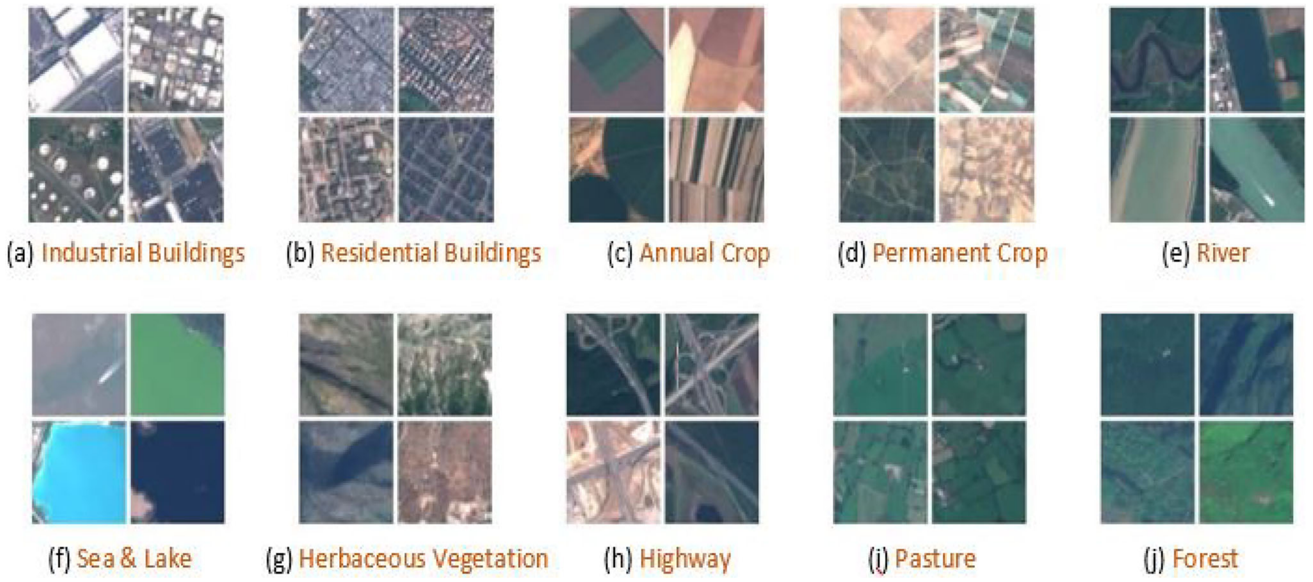


Fig. 8 Examples of EuroSAT dataset. It includes 27,000 images, and 2000–3000 images for each class

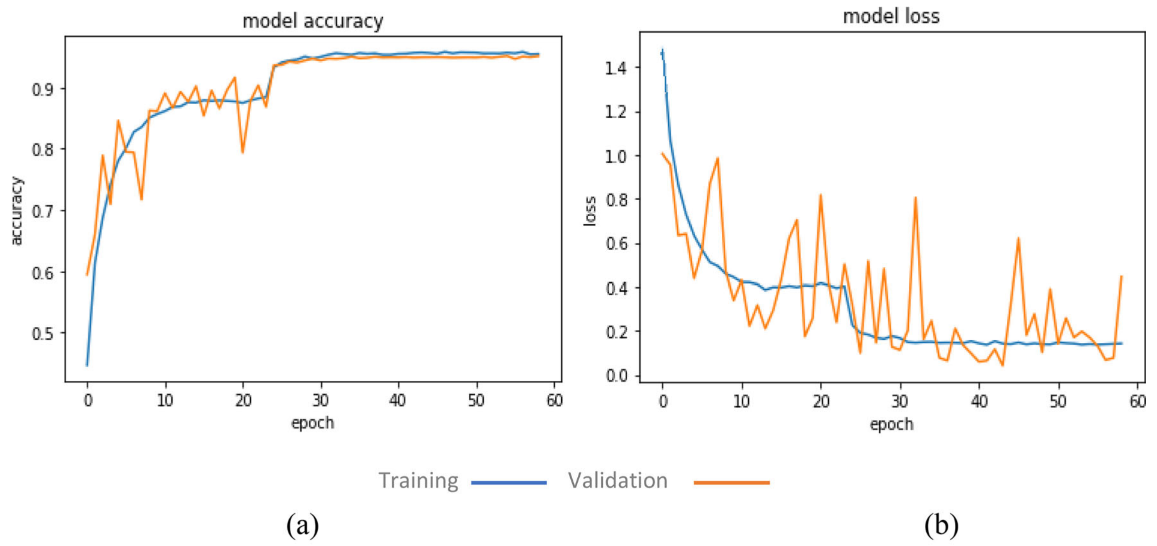


Fig. 9 The VGG16 results of training and validation, **a** accuracy and **b** loss during 60 epochs. Both the loss and the accuracy improved exponentially in the first epoch and subsequently exhibited a linear relation in epochs 2–23. Learning instability was observed during this period and significant improvements towards the end

Table 1 Performance of the deep convolutional SNN on the EuroSAT dataset

Class	Accurate prediction	Label count	Class accuracy
Annual crop	1370.0	1500	0.9360
Forest	1497.0	1500	0.9680
Herbaceous vegetation	1396.0	1500	0.9360
Highway	1186.0	1250	0.9232
Industrial	1195.0	1250	0.9856
Pasture	914.0	1000	0.9090
Permanent crop	1166.0	1250	0.9112
Residential	1498.0	1500	0.9420
River	1172.0	1250	0.9688
Sea/lake	1362.0	1500	0.9940

4.3 Performance

After training the deep convolutional model, we convert it into an SNN model by replacing the ReLU functions with spiking activations. Finally, we transfer the trained weights to a network of IF units and apply the weight–threshold balancing technique. The best overall model, which uses all 13 bands, accurately classified 94.88% of the testing set over $T = 400$ timesteps. Figure 9 shows the results of VGG16 (training and validation). Tables 1 and 2 show the corrected prediction result for each class and accuracy of Deep CNN and Deep CSNN models, respectively.

The training stopped at the 57th epoch due to an early stopping strategy. This method prevented the model from overfitting and saved computational time. Based on Table 1, the sea/lake, river, industrial, and forest classes showed the best performance above 96% accuracy on test dataset.

5 Second case study: epileptic seizure detection

One of the most common neurological disorders in the world is epilepsy, which affects people of all ages [55]. Thus, it is critical to suggest a highly precise and energy-efficient method of predicting seizure onsets. Electroencephalography (EEG) is a technique used to record electrical activity in the brain. An epileptic patient's EEG recordings can be classified into four states: interictal (between seizures), preictal (pre-seizures), ictal (during seizures), and postictal (post-seizures). The predictive task's goal is to distinguish the pre-seizure state from the other three states. However, this is a time-consuming and costly task even for a single patient. For this reason, recent research focuses on machine learning techniques for EEG signal processing such as CNN. However, CNNs are known to be computationally intensive, which makes them difficult to implement on wearable medical devices. Alternatively, SNN is an energy-efficient candidate for choice for embedded hardware. This section aims to deploy a deep convolutional SNN for epileptic seizure detection using a conversion method.

5.1 Methods

In this research, we propose a deep convolutional SNNs to make epileptic seizure predictions from the EGG samples. The proposed approach consists of the same procedure discussed in Sect. 4. Firstly, a number of modifications have been done to make the VGG16 model transferable to spiking model. After training the model with the ReLU activations function and getting the weights, the model transformed into a spiking domain by replacing the ReLU units with IF units. We also applied a weight–threshold balancing technique to ensure an accurate translation of continuous-valued neuron activations into the firing rates of a spiking neuron. Because the SNN layers only received binary spikes as inputs, the numerical EEG signals must be represented as spikes. We employ rate coding, which encodes each signal value normalised to (0, 1) as a spike train of length T. Figure 10 shows the conversion process of the Deep CNN to Deep SCNN for epileptic seizure prediction.

5.2 Dataset

We used the EEG recordings provided by the children's Hospital of Boston—Massachusetts Institute of Technology (CHB-MIT) to train and test the model for seizure prediction [56]. It includes 23 individuals and 969 h of scalp EEG recordings. The EEG recordings are divided into 24 sets and were recorded with a 16-bit resolution. Table 3 shows the details of the CHB-MIT EEG dataset.

Each patient's data is divided into 70/15/15 ratios for training, validation, and test, respectively. We need to mention the

Table 2 Accuracy of Deep CNN and Deep CSNN models

Network architecture	Spiking neuron model	ANN Acc (%)	SNN Acc (%)	Conversion loss (%)
VGG16	IF (rate-based)	95.36	94.88	0.48

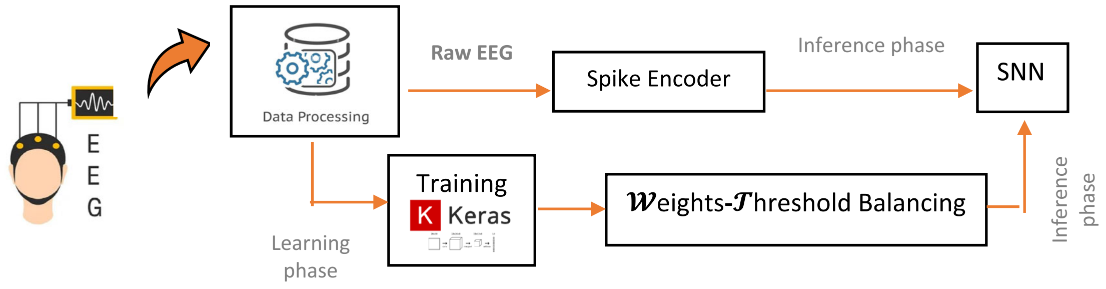


Fig. 10 Conversion process of the Deep CNN to Deep CSNN for epileptic seizure prediction

Table 3 Specifications of the CHB-MIT datasets

Dataset	Num of seizures	Num of patients	Sampling frequency (HZ)
CHB-MIT (EEG)	198	23	256

Table 4 Sensitivity and specificity of Deep CNN and Deep CSNN models

	ANN (%)	SNN (%)	Conversion loss (%)
Average sensitivity	96.4	95.06	1.34
Average specificity	99.72	99.45	0.27

raw recorded signals used in this work, as they need less computational resources from a hardware implementation point of view.

5.2.1 Data pre-processing

Imbalance number of sample in each class is one of the common problems in most classification datasets. This problem can also be found in the CHB-MIT, where only 198 seizures are available, suffering from class imbalances and few positive samples. To address this issue, we divided the EEG records into overlapping time windows with 50% overlap and 4-s window size. To get more samples from the seizure class, we increase the number of windows.

5.3 Performance

To evaluate the performance of Deep CSNN classifier, we used two well-known metrics, namely sensitivity and specificity. Sensitivity is the probability of seizure samples that the model correctly classifies as a seizure. Specificity is the proportion of non-seizure samples that the model correctly classifies as non-seizure.

The patient-specific Deep CNN algorithm was evaluated on the CHB-MIT database. Our results reveal that the sensitivity is 96.4% and the specificity is 99.66%. After converting Deep CNN to Deep CSNN, model was evaluated on test dataset. Our results reveal that the sensitivity is 95.06% and the specificity is 99.45% with $T = 200$. Figure 11 shows the sensitivity and specificity of the Deep CSNN model on test dataset. Table 4 shows the average sensitivity and specificity of Deep CNN and Deep CSNN models.

6 Discussion

In the first case study, deep convolutional SNNs were used to tackle the problem of LULC classification. For this task, a well-known learning architecture, namely, VGG16 was employed on the EuroSAT dataset. After training the VGG16 model, we transform it into an SNN by replacing the ReLU units with IF units. We transfer the trained weights to a network of IF units and apply weight–threshold balancing technique. The best overall model was able to accurately predict the classification for 94.88% of the testing set. The performance of the introduced model compared with different existing methods is presented in Table 5.

Dewangkoro and Arymurthy [57] combined CNN with Channel Squeeze & Spatial Excitation (sSE) block for LULC classification. The sSE block is used in their design to recalibrate the CNN function. They also used support vector machine (SVM) and Twin SVM (TWSVM) instead of the SoftMax classifier. Their three models (ResNet50, InceptionV3, and VGG19) accurately predicted the classification for 64.32%, 80.36%, and 94.57% of the testing set, respectively. Sonune [58] tackled the problem of land use and land cover classification with two models, namely random forest and ResNet-50. The authors achieved an accuracy of 94.25% and 61.46% using a ResNet50 and random forest model, respectively. In another work, Chen et al. [60] applied the knowledge distillation training approach for remote sensing scene classification on the EuroSAT dataset. Their experimental results showed overall an accuracy of 94.74%.

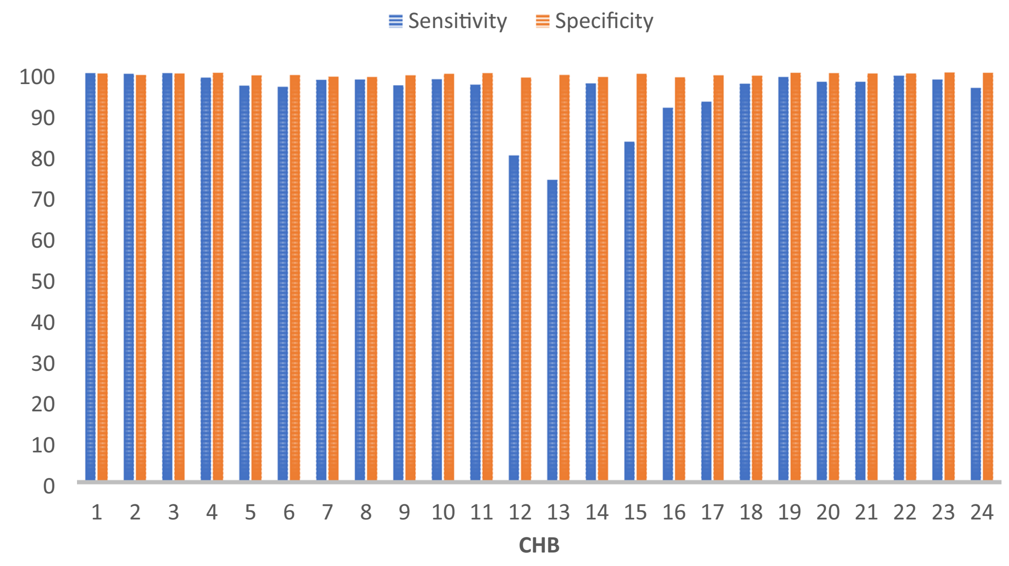


Fig. 11 Specificity and sensitivity of the Deep CSNN

Table 5 Performance of the introduced model compared to different methods

Reference	Model	Dataset	Accuracy (%)
This work	Deep convolutional SNN (VGG 16)	EuroSAT 80/10/10	94.88
Dewangkoro and Arymurthy [57]	ResNet50 + sSE + SVM	EuroSAT 70/30	64.32
	InceptionV3 + SVM		80.36
	VGG19 + sSE + TWSVM		94.57
Sonune [58]	Random forest	EuroSAT 80/20	61.46
	ResNet-50		94.25
	VGG19		97.66
Senecal et al. [59]	SpectrumNet w/standard convolution	EuroSAT NA	92.01
Chen et al. [60]	Knowledge distillation	EuroSAT 50/25/25	94.74

For the second case study, we have presented a deep convolutional SNNs for seizure detection. We used the overlapping windows method to address the problem of imbalanced datasets. As it is shown in Table 6, our model performance is comparable to state-of-the-art methods, with a sensitivity of 95.06% and a specificity of 99.45%. In the spiking domain, Tian et al. [61] proposed a Spiking-CNN to make seizure prediction. Their experiments only consider lead seizures that occur at least 4 h after the previous seizure, which means seven subjects in the CHB-MIT dataset. According to their experimental results, the sensitivity and specificity of 95.1% and 99.2% are achieved. Zhao et al. [62] proposed an FT-VGG16 classifier to classify seizure and non-seizure EEG signals. Their experimental results reveal the sensitivity and specificity of 98.75% and 98.18%, respectively. Several researchers applied SVM methods for epilepsy classification [63]. For example, Song et al. [64] used SVM and the weighted-permutation entropy method. They achieved an epilepsy detection accuracy of 91.62 for six different cases.

Considering the event-driven computational and resource-consuming benefits of the spiking neuron model, SNNs are promising for low-power embedded applications compared to conventional ANNs. To verify the energy consumption of our developed DCSNN, a neuromorphic hardware, like Loihi [66] or SpiNNaker [67], is needed as it has non-von Neumann architecture. Here, however, we provide an insight into energy consumption by evaluating the computational complexity of our DCSNN and against DCNN.

We employ the multiply-and-accumulate (MAC) operation to determine the computational complexity. Due to the use of different neuron models, the number of operations for the DCSNN and the DCNN is calculated differently. The computation cost of the DCNN is generally defined by the MAC procedure in Eq. (1), Sect. 2.2. Whereas in the DCSNN, data transferred between layers is a binary spike, and it performs asynchronous accumulate (AC) operations to update the state of each neuron in Eq. (2), Sect. 2.3.

We need to mention that the integration is event-driven in spiking neurons, indicating no computation occurs, if there is no spike delivered. The computation cost can be calculated

Table 6 Performance of the introduced patient-specific seizure detection model compared to different methods

Authors	Model	Sensitivity%	Specificity%	Dataset
This work	Deep convolutional	96.4	99.66	CHB-MIT
	Deep convolutional SNNs	95.06	99.45	
Tian et al. [61]	S-CNNs	95.1	99.2	CHB-MIT (Patients 1, 5, 6, 8, 10, 14, 22)
Gao et al. [65]	DCNNs	92.6	97.1	CHB-MIT
Zhao et al. [62]	FT-VGG16 and CWT	98.75	98.18	CHB-MIT
Karimi and Kassiri [53]	RBF-SVM	96.87	99.95	CHB-MIT
Song et al. [64]	SVM	95.83	96.15	CHB-MIT

as follow:

$$C_{\text{ANN}} = C_{\text{mul}} + C_{\text{add}} \tag{10}$$

$$C_{\text{SNN}} = C_{\text{add}} \tag{11}$$

where C_{add} is the computation cost of additions and C_{mul} is the computation cost for multiplications.

Although the DCSNN is required to evaluate over timesteps, its computation cost is lower than the DCNN, because the AC operations cost less than MAC operations. According to the findings in [66], an energy cost for a 32-bit AC operation is 5.1× more energy-efficient than a MAC operation. AC operation needs 0.9 pJ, whereas a MAC operation needs 4.6 pJ for 32-bit floating-point computation (45nm CMOS technology).

We have calculated the number of AC and MAC operations and energy consumption in 45nm technology, proposed by Horowitz [68]. Tables 7 and 8 compare the DCNN and DCSNN for patient-specific seizure detection, and LULC classification in terms of their energy cost with 50 timesteps, respectively.

We can see that the DCSNN has more addition operation than the DCNN, because of the rate coding time window. On the other hand, removing costly multiplication operations is helpful for energy reduction. According to calculations of energy consumption, the ratio of SNN energy to ANN energy for patient-specific seizure detection is 1.88 lower and the ratio for LULC classification is 1.86.

An important characteristic of SNNs is the time required to generate output. If the number of timesteps is too few, the SNN will not get enough information for training or inference. By contrast, a large number of timesteps results in an increase in prediction latency. Another observation is that a longer time window can result in improved accuracy but increases computing costs and energy consumption. As an example, Fig. 12 illustrates the comparison of LULC classification accuracy with different timesteps and its associated energy consumption.

As we expected, higher timestep leads to higher accuracy. However, this causes higher energy consumption. The timestep of 400 is sufficient to achieve 94.88 accuracy, while the lowest accuracy is obtained with a timestep of 50. Hence, having a trade-off between accuracy, inference latency, and energy remains a significant challenge in neuromorphic computing. One of the main reasons for this issue is the difficulty in determining proper hyperparameter values for the SNNs.

Future works will include applying a multi-objective optimization method to produce high-accuracy and low-latency inference. Moreover, we plan to continue this work by deploying the models on neuromorphic hardware, such as Loihi or SpiNNaker, enabling us to perform more accurate measurements and checking of the actual energy consumption of the SNNs.

7 Conclusion

This study presents an investigation on developing a deep convolutional spiking neural network for energy-efficient land cover and land use classification and epileptic seizure prediction. We convert pre-trained VGG16 into corresponding spiking equivalents with nearly comparable performance to the original one. We have tested the model datasets: EuroSAT dataset and CHB-MIT dataset. Experimental results show a classification accuracy of 94.88%, and seizure detection specificity of 99.45% and a sensitivity of 95.06%. Deep convolutional SNN training with a conversion method is promising and confirms that the benefits of DNNs are available for SNNs.

Table 7 The energy consumption comparison between DCNN and DCSNN for patient-specific seizure detection

Model	Task	Dataflow	Energy consumption In 45 nm technology	$\frac{SNN}{ANN}$
DCNN	Patient-specific seizure detection	32-bit float value	0.0993e−3J	1.88
DCSNN		Single bit of 0 or 1	0.0526e−3J	

Table 8 The energy consumption comparison between DCNN and DCSNN for LULC classification

Model	Task	Dataflow	Energy consumption In 45 nm technology	$\frac{SNN}{ANN}$
DCNN	LULC classification	32-bit float value	30.95e−3J	1.86
DCSNN		Single bit of 0 or 1	16.57e−3J	

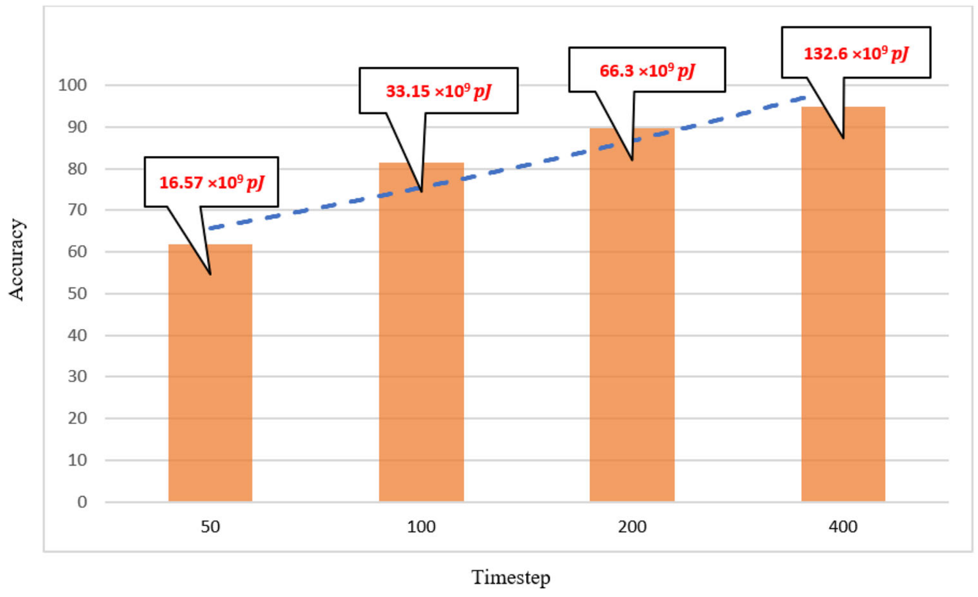


Fig. 12 The trade-off between accuracy, power consumption, and inference timesteps. The x-axis is SNN inference latency, the y-axis on the left measures the SNN inference accuracy, and the data label is estimated energy consumption based on 45 nm CMOS technology

Acknowledgements Not applicable.

Authors' contributions AJ, TTN, MAPM, and AZK were involved in conceptualization; AJ, TTN, MAPM, and AZK helped in methodology; AJ, TTN, MAPM, and AZK contributed to investigation; AJ helped in implementation, validation, writing—original draft preparation; AJ, TTN, MAPM, and AZK were involved in writing—review and editing; Supervision was done by TTN, MAPM, and AZK; AZK assisted in project administration.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions.

Declarations

Conflict of interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. C Zhang Y Lu 2021 Study on artificial intelligence: the state of the art and future prospects J. Ind. Inf. Integr. 23 100224
2. Shinde, P.P., Shah, S.: A review of machine learning and deep learning applications. In: 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), pp. 1–6 (2018)
3. Strubell, E., Ganesh, A., McCallum, A.: Energy and policy considerations for deep learning in NLP. arXiv preprint arXiv:1906.02243 (2019)

4. A Tavanaei M Ghodrati SR Kheradpisheh T Masquelier A Maida 2019 Deep learning in spiking neural networks Neural Netw. 111 47 63
5. A Calimera E Macii M Poncino 2013 The human brain project and neuromorphic computing Funct. Neurol. 28 3 191
6. A Belatreche 2013 Biologically Inspired Neural Networks OmniScriptum Publishing Saarbrücken
7. L Deng H Tang K Roy 2021 Understanding and bridging the gap between neuromorphic computing and machine learning Front. Comput. Neurosci. 15 665662
8. Q Fu H Dong 2021 An ensemble unsupervised spiking neural network for objective recognition Neurocomputing 419 47 58
9. M Mirsadeghi M Shalchian SR Kheradpisheh T Masquelier 2021 STiDi-BP: spike time displacement-based error backpropagation in multilayer spiking neural networks Neurocomputing 427 131 140
10. Qi, Y., Zhang, B., Taha, T. M., Chen, H., Hasan, R.: FPGA design of a multicore neuromorphic processing system. In: NAECON IEEE National Aerospace and Electronics Conference, pp. 255–258 (2014)
11. Q Xu J Peng J Shen H Tang G Pan 2020 Deep CovDenseSNN: A hierarchical event-driven dynamic framework with spiking neurons in noisy environment Neural Netw. 121 512 519
12. Fang, W., Yu, Z., Chen, Y., Masquelier, T., Huang, T., Tian, Y.: Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 2661–2671 (2021)
13. Ma, C., Xu, J., Yu, Q.: Temporal dependent local learning for deep spiking neural networks. In: 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1–7 (2021)
14. A Kugele T Pfeil M Pfeiffer E Chicca 2020 Efficient processing of spatio-temporal data streams with spiking neural networks Front. Neurosci. 14 439
15. Y Wu L Deng G Li J Zhu L Shi 2018 Spatio-temporal backpropagation for training high-performance spiking neural networks Front. Neurosci. 12 331
16. Wu, Y., Deng, L., Li, G., Zhu, J., Xie, Y., Shi, L.: Direct training for spiking neural networks: faster, larger, better. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33(01), pp. 1311–1318 (2019)
17. Kheradpisheh, S.R., Mirsadeghi, M., Masquelier, T.: BS4NN: Binarized spikingneural networks with temporal coding andlearning. Neural Process. Lett. **54**, 11255–1273 (2021)
18. Syed, T., Kakani, V., Cui, X., Kim, H.: Spiking neural networks using backpropagation. In: 2021 IEEE Region 10 Symposium (TENSYMP), pp. 1–5 (2021)
19. Wang, S., Li, C.: A supervised learning algorithm to binary classification problem for spiking neural networks. In: 2021 8th International Conference on Information, Cybernetics, and Computational Social Systems (ICCSS), pp. 448–452 (2021)
20. Yin, B., Corradi, F., Bohte, S.M.: Accurate online training of dynamical spiking neural networks through Forward Propagation Through Time. arXiv preprint arXiv:2112.11231 (2021)
21. Garg, I., Chowdhury, S. S., Roy, K.: DCT-SNN: using DCT To distribute spatial information over time for low-latency spiking neural networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4671–4680 (2021)
22. Li, Y., Guo, Y., Zhang, S., Deng, S., Hai, Y., Gu, S.: Differentiable Spike: rethinking gradient-descent for training spiking neural networks. In: Advances in Neural Information Processing Systems, vol. 34 (2021)
23. Wu, H., Zhang, Y., Weng, W., Zhang, Y., Xiong, Z., Zha, Z. J., et al.: Training spiking neural networks with accumulated spiking flow. In: Proceedings of the AAAI Conference on Artificial Intelligence. Virtual Event (2021)
24. Yao, M., Gao, H., Zhao, G., Wang, D., Lin, Y., Yang, Z., Li, G.: Temporal-wise attention spiking neural networks for event streams classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10221–10230 (2021)
25. Zheng, H., Wu, Y., Deng, L., Hu, Y. and Li, G.: Going deeper with directly trained larger spiking neural networks. arXiv preprint arXiv:2011.05280 (2020)
26. Eberlein, M., Hildebrand, R., Tetzlaff, R., Hoffmann, N., Kuhlmann, L., Brinkmann, B., Müller, J.: Convolutional neural networks for epileptic seizure prediction. In: 2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 2577–2582 (2018)
27. ND Truong AD Nguyen L Kuhlmann MR Bonyadi J Yang S Ippolito O Kavehei 2018 Convolutional neural networks for seizure prediction using intracranial and scalp electroencephalogram Neural Netw. 105 104 111
28. Xu, Y., Yang, J., Zhao, S., Wu, H., Sawan, M.: An end-to-end deep learning approach for epileptic seizure prediction. In: 2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), pp. 266–270 (2020)
29. F Achilles F Tombari V Belagiannis AM Loesch S Noachtar N Navab 2018 Convolutional neural networks for real-time epileptic seizure detection Comput. Methods Biomech. Biomed. Eng. Imaging Vis. 6 3 264 269
30. Antoniades, A., Spyrou, L., Took, C.C., Sanei, S.: Deep learning for epileptic intracranial EEG data. In: 2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP), pp. 1–6 (2016)
31. R Naushad T Kaur E Ghaderpour 2021 Deep transfer learning for land use and land cover classification: a comparative study Sensors 21 23 8083
32. G Furano G Meoni A Dunne D Moloney V Ferlet-Cavrois A Tavoularis 2020 Towards the use of artificial intelligence on the edge in space systems: challenges and opportunities IEEE Aerosp. Electron. Syst. Mag. 35 12 44 56
33. I Kiral-Kornek S Roy E Nurse B Mashford P Karoly T Carroll 2018 Epileptic seizure prediction using big data and deep learning: toward a mobile system EBioMedicine 27 103 111
34. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 25 (2012)
35. Y LeCun L Bottou Y Bengio P Haffner 1998 Gradient-based learning applied to document recognition Proc. IEEE 86 11 2278 2324
36. A Javanshir TT Nguyen MP Mahmud AZ Kouzani 2022 Advancements in algorithms and neuromorphic hardware for spiking neural networks Neural Comput. 34 6 1289 1328
37. R Brette 2015 Philosophy of the spike: rate-based versus spike-based theories of the brain Front. Syst. Neurosci. 9 151
38. Thorpe, S., Gautrais, J.: Rank order coding. In: Computational Neuroscience. Springer, Boston, pp. 113–118 (1998)
39. Kiselev, M.: Rate coding vs. temporal coding-is optimum between? In: 2016 international joint conference on neural networks (IJCNN), pp. 1355–1359 (2016)
40. JA Pérez-Carrasco B Zhao C Serrano B Acha T Serrano-Gotarredona S Chen B Linares-Barranco 2013 Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate coding and coincidence processing–application to feedforward ConvNets IEEE Trans. Pattern Anal. Mach. Intell. 35 11 2706 2719
41. N Abderrahmane E Lemaire B Miramond 2020 Design space exploration of hardware spiking neurons for embedded artificial intelligence Neural Netw. 121 366 386
42. AN Burkitt 2006 A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input Biol. Cybern. 95 1 1 19
43. Diehl, P.U., Neil, D., Binas, J., Cook, M., Liu, S.C., Pfeiffer, M.: Fast-classifying, high-accuracy spiking deep networks through

weight and threshold balancing. In: 2015 International joint conference on neural networks (IJCNN), pp. 1–8 (2015)
44. Diehl, P.U., Pedroni, B.U., Cassidy, A., Merolla, P., Neftci, E., Zarrella, G.: Truehappiness: neuromorphic emotion recognition on truenorth. In: 2016 International Joint Conference on Neural Networks (IJCNN), pp. 4278–4285 (2016)
45. Cassidy, A.S., Merolla, P., Arthur, J.V., Esser, S.K., Jackson, B., Alvarez-Icaza, R. et al.: Cognitive computing building block: a versatile and efficient digital neuron model for neurosynaptic cores. In: The 2013 International Joint Conference on Neural Networks (IJCNN), pp. 1–10 (2013)
46. B Rueckauer IA Lungu Y Hu M Pfeiffer SC Liu 2017 Conversion of continuous-valued deep networks to efficient event-driven networks for image classification Front. Neurosci. 11 682
47. Y Cao Y Chen D Khosla 2015 Spiking deep convolutional neural networks for energy-efficient object recognition Int. J. Comput. Vis. 113 54 66
48. Pirson, T., Bol, D., Frenkel, C.: Training ultra-low-power spiking neural networks for neuromorphic IoT vision sensing and recognition (Doctoral dissertation, Master's thesis, Ecole Polytechnique de Louvain, Université Catholique de Louvain, 2019. Prom. Bol, David
49. Eshraghian, J.K., Ward, M., Neftci, E., Wang, X., Lenz, G., Dwivedi, G., et al.: Training spiking neural networks using lessons from deep learning. arXiv preprint arXiv:2109.12894 (2021)
50. EO Neftci C Augustine S Paul G Detorakis 2017 Event-driven random backpropagation: enabling neuromorphic deep learning machines Front. Neurosci. 11 324
51. Y Cao Y Chen D Khosla 2015 Spiking deep convolutional neural networks for energy-efficient object recognition Int. J. Comput. Vis. 113 1 54 66
52. A Sengupta Y Ye R Wang C Liu K Roy 2019 Going deeper in spiking neural networks: VGG and residual architectures Front. Neurosci. 13 95
53. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations, ICLR, San Diego, (2015)
54. P Helber B Bischke A Dengel D Borth 2019 Eurosat: a novel dataset and deep learning benchmark for land use and land cover classification IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 12 7 2217 2226
55. Kwan, P., Arzimanoglou, A., Berg, A.T., Brodie, M.J., Allen Hauser, W., Mathern, G., et al.: Definition of drug resistant epilepsy: consensus proposal by the ad hoc Task Force of the ILAE Commission on Therapeutic Strategies (2010)
56. AL Goldberger LA Amaral L Glass JM Hausdorff PC Ivanov RG Mark 2000 PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals Circulation 101 23 e215 e220
57. Dewangkoro, H.I., Arymurthy, A.M.: Land use and land cover classification using CNN, SVM, and channel squeeze & spatial excitation block. In: IOP Conference Series: Earth and Environmental Science, vol. 704 (1), p. 012048 (2021)
58. Sonune, N.: Land Cover Classification with EuroSAT Dataset. 2020. Available online: https://www.kaggle.com/nilesh789/landcover-classification-with-eurosat-dataset
59. Senecal, J.J., Sheppard, J.W., Shaw, J.A.: Efficient convolutional neural networks for multi-spectral image classification. In: 2019 International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2019)
60. G Chen X Zhang X Tan Y Cheng F Dai K Zhu 2018 Training small networks for scene classification of remote sensing images via knowledge distillation Remote Sens 10 5 719
61. Tian, F., Yang, J., Zhao, S., Sawan, M.: A new neuromorphic computing approach for epileptic seizure prediction. In: 2021 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1–5 (2021)
62. W Zhao W Zhao W Wang X Jiang X Zhang Y Peng B Zhang G Zhang 2020 A novel deep neural network for robust detection of seizures using EEG signals Comput. Math. Methods Med. https://doi.org/10.1155/2020/9689821
63. Karimi, M. R., Kassiri, H.: A multi-feature nonlinear-SVM Seizure detection algorithm with patient-specific channel selection and feature customization. In: 2020 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1–5 (2020)
64. Song, Z., Wang, J., Cai, L., Deng, B., Qin, Y.: Epileptic seizure detection of electroencephalogram based on weighted-permutation entropy. In: 2016 12th World Congress on Intelligent Control and Automation (WCICA), pp. 2819–2823 (2016)
65. Y Gao B Gao Q Chen J Liu Y Zhang 2020 Deep convolutional neural network-based epileptic electroencephalogram (EEG) signal classification Front. Neurol. 11 375
66. M Davies N Srinivasa TH Lin G Chinya Y Cao SH Choday 2018 Loihi: a neuromorphic manycore processor with on-chip learning IEEE Micro 38 1 82 99
67. S Furber 2016 Large-scale neuromorphic computing systems J. Neural Eng. 13 5 051001
68. Horowitz, M.: 1.1 computing's energy problem (and what we can do about it). In: 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), pp. 10–14 (2014)