ECG Classification Algorithm Based on STDP and R-STDP Neural Networks for Real-Time Monitoring on Ultra Low-Power Personal Wearable Devices

Alireza Amirshahi and Matin Hashemi, *Member, IEEE*

***Abstract*—This paper presents a novel ECG classification algorithm for inclusion as part of real-time cardiac monitoring systems in ultra low-power wearable devices. The proposed solution is based on spiking neural networks which are the third generation of neural networks. In specific, we employ spike-timing dependent plasticity (STDP), and reward-modulated STDP (R-STDP), in which the model weights are trained according to the timings of spike signals, and reward or punishment signals. Experiments show that the proposed solution is suitable for real-time operation, achieves comparable accuracy with respect to previous methods, and more importantly, its energy consumption in real-time classification of ECG signals is significantly smaller. In specific, energy consumption is 1.78 μJ per beat, which is 2 to 9 orders of magnitude smaller than previous neural network based ECG classification methods.**

***Index Terms*—Cardiac monitoring, Electrocardiogram (ECG) classification, embedded real-time systems, low power consumption, machine learning, spiking neural network (SNN), wearable devices.**

I. Introduction

CARDIOVASCULAR diseases (CVDs) account for a large proportion of global deaths. Early detection, as in many other diseases, plays a crucial role in the process of stopping or controlling the progression of CVDs. Cardiac arrhythmias are widely used as signs for many CVDs. Since the arrhythmias are reflected into electrical activities of the heart, they can be detected by analyzing Electrocardiogram (ECG) signals.

Visual inspection of recorded ECG signals during a visit to the cardiologist is the traditional form of arrhythmia detection. However, due to their intermittent occurrence, specially in early stages of the problem, arrhythmias are difficult to detect from a short time window of the ECG signal. Therefore, continuous ECG monitoring is crucial in early detection of potential problems [1].

In recent years, personal wearable devices have been introduced as cost-effective solutions for continuous ECG monitoring in daily life. Previous works include ECG monitoring solutions that are low-power but only extract the R peak or the QRS complex and do not further process the ECG signal [2]–[5]. Many others detect arrhythmia patterns such as premature ventricular contraction and ventricular escape beats by fully analyzing the ECG signal via classification algorithms, but transmit the signal to a connected smartphone or a remote cloud server for performing the heavy computations associated with ECG classification algorithms [6]. Besides privacy concerns [7], employing such solutions for continuous cardiac monitoring is limited by the availability, speed and energy consumption of the wireless connection. Other solutions include recording ECG signals along with other vital signals through wireless devices [8], [9], and offline analysis of the recorded signals [10], [11]. Our approach, on the other hand, is to continuously monitor the ECG signal in real-time and on the wearable device itself.

In this approach, the ECG signal is continuously monitored without any connection to or any assistance from other systems. However, the main limiting factor is that wearable health monitoring devices should be small, and thus, may not utilize powerful processors or large batteries. Therefore, computational costs and energy consumption of ECG classification algorithms are important for real-time operation on small and low-power wearable devices [1].

Classical algorithms were mainly based on morphological features, such as the QRS complex, and signal processing techniques such as Fourier transform and wavelet transform [12]–[15]. Such features show significant variations among different individuals and under different conditions, and therefore, are not sufficient for accurate arrhythmia detection [16], [17]. Many solutions have been proposed based on artificial neural networks, and especially in recent years, deep convolutional neural networks (CNN) and recurrent neural networks (RNN) [17]–[23]. Neural network algorithms automatically extract the features from data, and hence, provide higher accuracy. This is because neural network based feature extraction is inherently more resilient to variations among different ECG waveforms [17]. However, the downsides of employing deep neural networks are large power consumption and high computational intensity. Low-power embedded multiprocessors exist for real-time stream processing, but their efficient programming involves certain challenges [24]–[27].

Spiking neural networks (SNN) are the third generation of neural networks and provide the opportunity for ultra low-power

Manuscript received May 8, 2019; revised July 22, 2019 and September 4, 2019; accepted October 6, 2019. Date of publication October 22, 2019; date of current version December 31, 2019. This work was supported in part by Iran National Science Foundation under Grant 95835171. This paper was recommended by Associate Editor Y. Zheng. *(Corresponding author: Matin Hashemi.)*

The authors are with the Learning and Intelligent Systems Laboratory, Department of Electrical Engineering, Sharif University of Technology, Tehran 11356, Iran (e-mail: alireza.amirshahi@ee.sharif.edu; matin@sharif.edu).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TBCAS.2019.2948920

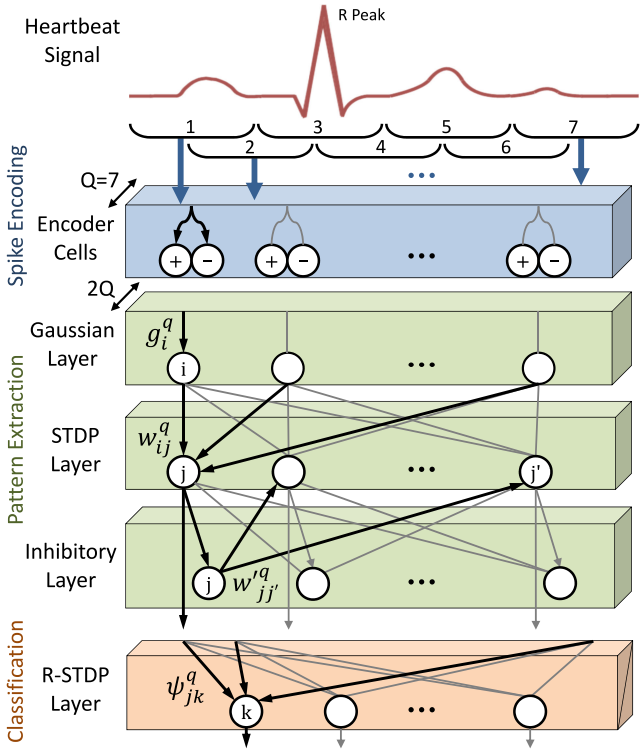

Fig. 1. Overall view of the proposed solution.

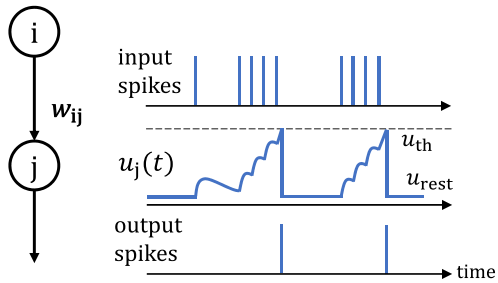

Fig. 2. In this example, neuron i sends nine spikes to neuron j. As a result, the membrane potential of neuron j, i.e., u_j, is increased and reached to u_{th} twice. Hence, two spikes are fired on its output.

operation [28]–[34]. Here the information is processed based on propagation of spike signals through the network as well as the timing of the spikes. Every neuron consumes energy only when necessary, that is only when it sends or receives spikes. In recent years, neuromorphic circuits have been introduced for ultra low-power implementation of SNN-based algorithms. Examples include IBM TrueNorth, Intel Loihi, and Qualcomm Zeroth.

This paper proposes a novel SNN-based ECG classification algorithm for real-time operation on ultra low-power wearable devices. The overall view of the proposed solution is shown in Fig. 1. The heartbeat signal is split into a small number of overlapping windows, which are then encoded into spike signals. The patterns in the generated spikes are automatically extracted in the STDP layer with assistance from a Gaussian layer and an inhibitory layer. Finally, the extracted features are classified in the R-STDP layer. STDP stands for spike-timing dependent plasticity which means the timings of the spikes are used to train the weights in this layer. R-STDP stands for reward-modulated STDP which means reward or punishment signals are used to train the weights, in addition to the spike timings. To improve classification performance, the learning rules in training STDP and inhibitory layers are optimized in order to better fit the ECG classification domain. Since all the above layers are performed in the spike domain, the energy consumption required for inference, i.e., real-time classification of ECG signals, is reduced significantly. Experiments show that the proposed solution is suitable for real-time operation, achieves comparable classification performance with respect to previous methods, and more importantly, its energy consumption for classification of ECG signals is multiple orders of magnitude smaller than previous neural network based methods.

Previous SNN-based solutions that process ECG signals employ non-SNN algorithms for pattern extraction [35], or do not fully analyze the ECG signal, for instance, only extract the heart rate [36], apply a band-pass filter to the input ECG signal [37], or detect perturbations in the signal [38].

It is noteworthy to mention that SNNs have previously been employed in many areas in computer vision. Examples include digit recognition and visual categorization [39]–[42]. This is mainly because the spiking neuron models are inspired from neuroscience studies on the brain's visual cortex. SNNs have also been applied to non-vision applications including robotics control [43] and classification of EEG spatio-temporal data [44].

The rest of this manuscript is organized as the following. Section II provides a brief background on the operational model and circuit implementation for spiking neurons. Section III presents our proposed SNN-based ECG classification algorithm. Section IV presents the results of experimental evaluations as well as comparisons with previous works in term of classification performance, network type, and energy consumption. Section V concludes the paper.

II. Spiking Neural Networks

A. Operational Model

In contrast to the second generation neural networks, such as multi-layer perceptrons, in the third generation neural networks, i.e., spiking neural networks, the concept of time is deeply incorporated into the operational model [45]. In specific, the *neurons* communicate through electrical signals called *spikes*. For example in Fig. 2, neuron i sends nine spikes to neuron j at different times. The information is encoded in the timing of the spikes, not in their amplitude.

A neuron i fires at time t, i.e., generates a spike on its output at time t, only when its *membrane potential* $u_i(t)$ reaches a specific threshold u_{th}. When neuron i fires, u_i is decreased to u_{rest} and prohibited from rising for a short time period called *refractory time*. The generated spike travels to neuron j and causes an increase in u_j, i.e., membrane potential of neuron j. The above procedure is shown in Fig. 2. Since neurons do not fire all the time, the energy consumption is significantly smaller than other types of neural networks [29]–[34].

A connection from neuron i to j is called a *synapse*. For this synapse, i and j are called *pre-synaptic* and *post-synaptic* neurons, respectively. A *Weight* w_{ij} is associated with every synapse ij.

Different brain-inspired mathematical models exist for SNNs. We employ one of the most popular models called Leaky Integrate-and-Fire (LIF) model [45] which operates based on the following equation. This differential equation describes how u_j is changed at time t.

$$\tau \frac{d}{dt} u_j(t) = -\big(u_j(t) - u_{\text{rest}}\big) + \alpha \sum_i s_i(t) w_{ij} \qquad (1)$$

The index i iterates through all neurons whose outputs are connected to neuron j. The term $s_i(t)$ is equal to either one or zero, and represents whether neuron i has generated a spike at time t. Hence, the term $\sum_i s_i(t) w_{ij}$ is zero when no spike is received by neuron j at time t. When one or multiple spikes are received at time t, the term $\sum_i s_i(t) w_{ij}$ causes an increase in $\frac{d}{dt} u_j(t)$. In addition, a leakage current causes membrane potential u_j to tend to the rest potential u_{rest} with time constant τ. Hence, the constant parameter α determines the strength of $\sum_i s_i(t) w_{ij}$ compared to the leakage current in changing $\frac{d}{dt} u_j(t)$. To summarize, u_j does not change at time t, if it is equal to its rest value u_{rest} and no spike is received. Otherwise, $\frac{d}{dt} u_j(t)$ is non-zero, and thus, $u_j(t)$ changes at time t.

According to the above operational model, synaptic weight w_{ij} impacts spike propagation from neuron i to j. A larger w_{ij} means a spike generated by neuron i is more likely to trigger a spike by neuron j. This is because $s_i(t) w_{ij}$ affects $\frac{d}{dt} u_j(t)$. As a result, the larger the weight w_{ij} the higher the firing rate of neuron j.

B. Circuit Implementation

Many circuits with various optimizations to reduce energy consumption have been proposed for implementing spiking neurons. Here we provide a brief introduction to a simplified circuit common to most ultra low-power implementations [29]–[34].

Fig. 3(a) shows a spiking neuron. The input current $I_i(t)$ represents the spike which enters the neuron from a connected input synapse. The main part of this circuit is the capacitor which holds the membrane potential $u_j(t)$. The transistors in the leakage part of the circuit are responsible for decreasing $u_j(t)$, at a rate proportional to τ, in absence of the input current. In the comparison part, the value of $u_j(t)$ is compared with the threshold u_{th} in order to generate spikes. When a spike is generated, it is sent to the output synapse, and charges the refractory capacitor. The purpose of the refractory and reset parts are to force $u_j(t)$ to have the resting potential u_{rest} for a short period. When the refractory capacitor discharges, the reset transistor is turned off, and the membrane potential $u_j(t)$ may start to rise again by the input current.

Fig. 3(b) represents a differential-pair integrator synapse circuit. A spike generated by a pre-synaptic neuron turns the input switch on. When a spike is received, the synapse capacitor discharges at a rate proportional to the weight w. When $u_{syn}(t)$ is decreased, the output transistor is turned on and provides the output current $I_i(t)$. Once the input spike ends, the capacitor charges again, the output transistor is turned off, and the output current $I_i(t)$ becomes zero.

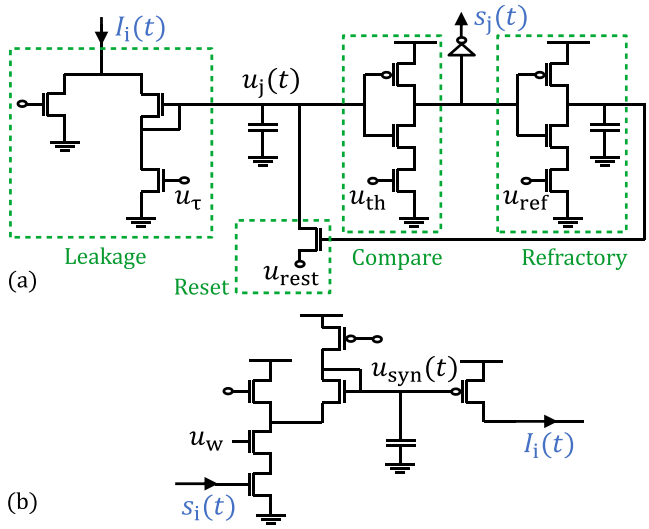

Fig. 3. A simplified circuit model for (a) spiking neuron j and (b) its input excitatory synapse ij coming from neuron i.

III. Proposed ECG Classification Algorithm

Our proposed SNN-based algorithm for classification of ECG beats is presented in this section. Fig. 1 illustrates an overall view of the proposed solution. The input ECG signal is first segmented into heartbeats. The heartbeat is split into a small number of windows, which are then encoded into spikes. The patterns in the generated spikes are automatically extracted in the STDP layer with assistance from two other layers, namely the Gaussian and the inhibitory layers. Finally, the extracted features are classified in the R-STDP layer. The formulations which govern the STDP and the inhibitory layers are optimized to better fit ECG classification problem. The details are discussed in the following.

A. Segmentation

Heartbeat segments are formed as 0.25 seconds of the input ECG signal before an R peak and 0.45 seconds after. R peak is a specific point in the ECG waveform as shown in Fig. 1. There exist many ultra low-power and highly accurate methods for R peak detection [3], [4].

Since the R peak has a much larger amplitude compared to other parts of the heartbeat, it becomes the dominant pattern in the STDP layer and basically causes the effect of other patterns to be highly reduced. In order to avoid this issue and efficiently capture all the patterns, every heartbeat is split into $Q = 7$ overlapping windows as shown in Fig. 1. The windows are processed separately in the pattern extraction step. Let's denote the heartbeat signal as X_{ecg} and the portion of X_{ecg} which falls in window $q \in [1, Q]$ as X^q_{ecg}. The number of samples in X^q_{ecg}

is given by the following equation.

$$|X^q_{ecg}| = \frac{1}{\lceil Q/2 \rceil} \times |X_{ecg}| \tag{2}$$

B. Spike Encoding

There is an encoder cell for every sample $i \in [1, |X^q_{ecg}|]$ from every window $q \in [1, Q]$. Every cell encodes its input into a series of spikes as the following. Spike generation is random and follows the Poisson process. The spike firing rate of this random Poisson process is set proportional to $X^q_{ecg}[i]$, i.e., the amplitude of sample i from window q, plus a small bias. The higher the amplitude, the higher the rate of spike firing of the corresponding cell [46].

In fact, since the amplitude can be both positive or negative, two encoder cells are employed for every sample, not one. Spikes are generated by the first encoder cell if the signal is positive, and by the second cell if negative. This is inspired by the nerve cells in the Retina [47].

The output of every encoder cell is connected to one synapse, and the generated spikes are fed into this synapse. Therefore, the total number of output synapses for a window $q \in [1, Q]$ is equal to

$$2 \times |X^q_{ecg}| \tag{3}$$

These synapses are split in two groups. As shown in Fig. 1, the number of windows in the next layers is equal to $2Q$. The positive encoder cells are connected to the synapses in the odd windows, and the negative encoder cells are connected to the synapses in the even windows.

C. Gaussian Layer

The purpose of this layer is to adjust the number of spikes in order to help the STDP-based pattern extraction that is discussed later in Section III-D.

Every window $q \in [1, 2Q]$ is processed separately as the following. As shown in Fig. 1, input synapse i in window q is connected to one neuron and its synaptic weight is set to g^q_i. As a result, the neuron adjusts the spike firing rate by a factor of g^q_i. In specific, we have

$$R^q_{\text{out},i} = g^q_i \times R^q_{\text{in},i} \tag{4}$$

where $R^q_{\text{in},i}$ and $R^q_{\text{out},i}$ denote the rate of spikes on the input and on the outputs of the neuron, respectively.

Since the windows overlap, an ECG peak which falls on the side of a window also appears in the middle of a neighbor window. In order to reduce the effect of side peaks, a Gaussian kernel is employed in setting the value of g^q_i. In specific, g^q_i is set as

$$g^q_i = \beta^q \times \frac{1}{\sigma\sqrt{2\pi}} \times e^{-\frac{1}{2}(\frac{i-\mu}{\sigma})^2} \tag{5}$$

where $\mu = \frac{1}{2}|X^q_{ecg}|$ and $\sigma = \frac{1}{3}|X^q_{ecg}|$. Note that $|X^q_{ecg}|$ denotes the window length. The mean μ is chosen such that the Gaussian function be centered at the middle of the window. The variance σ is chosen such that the effect of a side peak is reduced but not completely neglected.

The term β^q is a trainable variable which is different for different values of q, i.e., different windows. It is initialized as $\beta^q = 1$, and then trained as the following. The next layers operate more efficiently if all windows $q \in [1, 2Q]$ in this layer generate equal average number of spikes. Let $\overline{R^q_{\text{out}}}$ denote the average number of spikes fired by the neurons in window q in this layer. The larger the value of β^q, the larger the value of $\overline{R^q_{\text{out}}}$.

Therefore, in training β^q we would like to have $\overline{R^q_{\text{out}}}$ reach the same target rate R_{target} for all $q \in [1, 2Q]$. Since the neuron behavior (Section II) is non-linear, to achieve the desired rate, β^q is initialized to 1, then, the train data is fed to the network, and β^q is iteratively updated as the following.

$$\Delta\beta^q = \alpha \left(1 - \frac{\overline{R^q_{\text{out}}}}{R_{\text{target}}}\right) \tag{6}$$

D. STDP-Based Pattern Extraction

Spike-timing dependent plasticity (STDP) [48] is employed here to extract the patterns in the ECG signal. STDP is an unsupervised learning procedure. Every window $q \in [1, 2Q]$ is processed separately as the following. Let w^q_{ij} denote the synaptic weights in window q in this layer. The synaptic weights, also known as plasticity, are trained according to the timing patterns of the spikes in pre- and post-synaptic neurons. In specific, consider a synapse ij in this layer which connects pre-synaptic neuron i to post-synaptic neuron j. If the spike time of the pre-synaptic neuron, denoted as t_{pre}, is earlier than the spike time of the post-synaptic neuron, denoted as t_{post}, then the weight w_{ij} is increased. This is called long-term potentiation (LTP). Similarly, when t_{post} is earlier than t_{pre}, the weight is decreased. This is called long-term depression (LTD).

The smaller the time difference, the larger the amount of change in the synaptic weight. This is shown in Fig. 4(a). Here, $\Delta t = t_{\text{post}} - t_{\text{pre}}$, and $\Delta w = w_{\text{new}} - w_{\text{old}}$. As shown in the figure, $|\Delta w|$ is maximum when $|\Delta t| = 0$, and it is exponentially decreased with larger values of $|\Delta t|$. The mathematical equation for the STDP learning rule is

$$\Delta w = \begin{cases} a^+ \times e^{\left(-\frac{|\Delta t|}{\tau}\right)} & \Delta t > 0 \\ a^- \times e^{\left(-\frac{|\Delta t|}{\tau}\right)} & \Delta t < 0 \end{cases} \tag{7}$$

where, a^+ and a^- are learning rates, and τ is the time constant. Note that a^+ and a^- are positive and negative constant values, respectively.

Analysis of the Learning Rule: The learning rule in (7) creates a form of positive feedback which pushes the synaptic weights towards $+\infty$ or $-\infty$. The reason is described in the following. When a synaptic weight w_{ij} is increased, the term $s_i(t)w_{ij}$ in (1) will be larger for the next time, and therefore, the next spike from neuron i is more likely to cause a new spike by neuron j. This further increases w_{ij}. Since this positive feedback is likely to happen many times, the probability that this weight

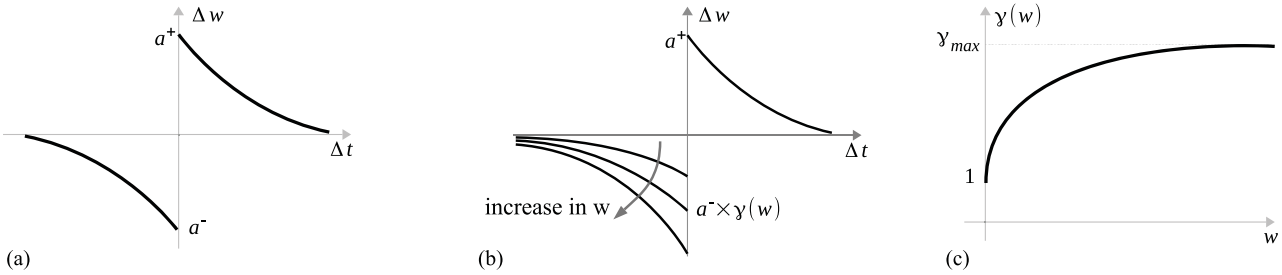

Fig. 4. Learning rule for the synaptic weights in the STDP layer. (a) The usual STDP learning rule. (b) Proposed learning rule. (c) The value of $\gamma(w)$ in (b) is between 1 and $\gamma_{\max}$.

is pushed towards $+\infty$ is very high. Similar situation happens in decreasing the weights, and thus, some of the weights are pushed towards $-\infty$. Note that spikes arrive randomly, and hence, during the STDP training, a weight is both increased and decreased many times, depending on the values of Δt. However, the net effect will be either positive or negative. Therefore, the weights are pushed towards $+\infty$ or $-\infty$.

Optimized Learning Rule: In computer vision applications, the negative weights are usually clipped to zero and the positive weights to a constant $w_{\max}$. The synaptic weights that are close or equal to $w_{\max}$ form the detected patterns in the input image [39]–[42]. In ECG classification, however, the clipping alone is not sufficient, and thus, we also optimize the STDP learning rule as

$$\Delta w = \begin{cases} a^+ \times e^{\left(-\frac{|\Delta t|}{\tau}\right)} & \Delta t > 0 \\ a^- \times \gamma(w) \times e^{\left(-\frac{|\Delta t|}{\tau}\right)} & \Delta t < 0 \end{cases} \tag{8}$$

where, $\gamma(w)$ is equal to

$$\gamma(w) = \frac{1 + w \cdot \gamma_{\max}}{1 + w} \tag{9}$$

The corresponding curves are shown in Fig. 4(b) and 4(c). The intuition behind this optimization is discussed in the following. Consider the toy example shown in Fig. 5, in which three pre-synaptic neurons i', i'', and i''' are connected to a post-synaptic neuron j. Consider the three input signals shown in Fig. 5. Neuron i' generates a very small number of spikes for all the three input signals. This is because the signal amplitude is zero, and hence, the encoder cell which is connected to neuron i' fires at a very low frequency. On the other hand, for all the three input signals, neuron i''' generates many spikes. This is because the signal amplitude is large, and hence, the encoder cell which is connected to neuron i''' fires at a high frequency. Since the generated spikes cause neuron j to fire as well, the synaptic weight $w_{i'''j}$ is increased many times. This also decreases $w_{i'j}$ because the spike frequency of i' is much lower, and Δt is negative in some cases. Therefore, in both the learning rules (7) and (8), $w_{i'j}$ and $w_{i'''j}$ are trained as 0 and $w_{\max}$, respectively. Basically, neuron j is trained to be sensitive to a zero amplitude at location i' and a large peak at location i''' in the input ECG signal.

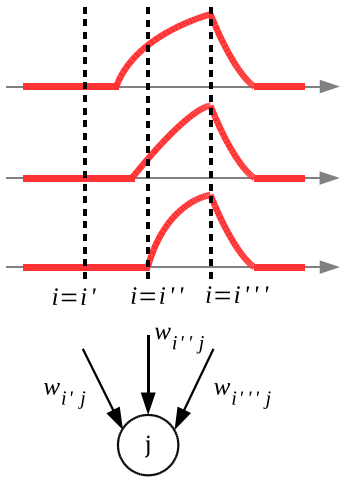

Fig. 5. Toy example to demonstrate the intuition behind optimized STDP learning rule in (8).

Now, let's study training of the synaptic weight $w_{i''j}$. Note that spike generation by the encoder cells follows a random process. Hence, it is possible that neuron i'' fires before i'''. In addition, note that the weights are initialized to random values. Hence, it is possible that $w_{i''j}$ be initialized to a larger value than $w_{i'''j}$. Therefore, the learning rule in (7) may train $w_{i''j}$ as $w_{\max}$. It is also possible, depending on the random settings, that $w_{i''j}$ be trained as 0. This random training to either zero or $w_{\max}$ is not desired because it lowers the capability of the network in capturing complex ECG patterns that have various amplitudes in different locations of the input signal.

The proposed learning rule in (8), however, trains $w_{i''j}$ differently. Since the three input signals in this example have different amplitudes at location i'', we would like $w_{i''j}$ to be trained as somewhere between 0 and $w_{\max}$. This is achieved in the proposed learning rule, because as w increases, $\gamma(w)$ is increased as well, i.e., LTD is amplified (Fig. 4(b)). Hence, although w is decreased less often than increased, i.e., the likelihood of negative Δt is smaller than positive Δt, the net effect of decreasing w is amplified by making LTD stronger than LTP. As a result, $w_{i''j}$ is trained as desired.

As shown in Fig. 4(c), the curve $\gamma(w)$ has a sharp slope near $w = 0$. Hence, LTP is amplified quickly with a small increase in w. This helps to suppress the effect of minor variations in the input ECG signal by lowering the probability that such variations

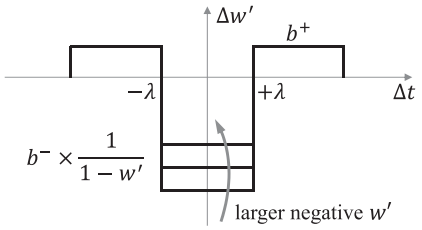

Fig. 6. Proposed learning rule for the backward synapses in the inhibitory layer. Note that w' is always kept negative.

can contribute to the captured patterns. This is because it quickly becomes difficult for them to be able to increase the weights. However, once a weight has been increased enough, i.e., a real pattern has been captured, the difficulty stays nearly the same, i.e., the curve $\gamma(w)$ does not increase much because it has a very small slope.

E. Inhibitory Layer

In order to prevent similar or same patterns from possessing the attention of all neurons in the STDP layer, inhibitory neurons are added to the network as shown in Fig. 1. Inhibitory layer makes the STDP layer more efficient in capturing different patterns [49].

As shown in Fig. 1, there is exactly one inhibitory neuron for every neuron in the STDP layer. When a neuron j in the STDP layer fires, its corresponding inhibitory neuron j fires as well and prevents the firing of all the other neurons $j' \neq j$ in the same window in the STDP layer. This is achieved through backward synapses with negative weights $w'^q_{jj'}$ which carry the fired spikes from inhibitory neuron j to all neurons $j' \neq j$ in the STDP layer and decrease their membrane potentials.

The negative synaptic weights $w'^q_{jj'}$ are trained as the following. When neurons from the same window in the STDP layer fire at about the same time, w' is decreased. The inhibitory learning rule is

$$\Delta w' = \begin{cases} b^- & |\Delta t| \leq \lambda \\ b^+ & |\Delta t| > \lambda \end{cases} \tag{10}$$

where, learning rates b^+ and b^- are positive and negative constant values, respectively. When spikes from neurons j and j' in the STDP layer are within λ time units from one another, w' is decreased by b^-. Otherwise, it is increased by b^+. Note that w' is clipped to zero, so it always holds a negative value [50].

Optimized Learning Rule: When the divergence among the existing patterns are small, the inhibition becomes too strong, which in turn decreases the number of spikes and incapacitates STDP training. To prevent this, we propose to modify the learning rule as shown in (11) and illustrated in Fig. 6.

$$\Delta w' = \begin{cases} b^- \times \dfrac{1}{1-w'} & |\Delta t| \leq \lambda \\ b^+ & |\Delta t| > \lambda \end{cases} \tag{11}$$

In the proposed learning rule, $\Delta w'$ is a smaller negative value as w' becomes a larger negative value. This modification guarantees that the negative weights do not grow too strong.

Asymmetric Structure: Our additional strategy for preventing the repercussion of sameness in the detected patterns is to employ an asymmetric structure in the inhibitory layer. In specific, in every epoch, some of the inhibitory neurons are randomly inactivated. This basically distributes the authority among the neurons in the STDP layer. The core idea is inspired by the dropout technique in deep convolutional neural networks [51]. Note that inhibitory neurons are only present during the training stage, and they are inactivated during the inference (test) stage.

F. Classification via R-STDP

This layer classifies the patterns which are previously extracted in the STDP layer. Here, reward-modulated STDP (R-STDP) is employed as the classifier. R-STDP operates not only based on spike timings but also a modulator signal. In the brain, the modulator signal can be a reward or punishment depending on Dopamine secretion [52].

As shown in Fig. 1, in our proposed solution, R-STDP is used for training synaptic weights ψ^q_{jk} in the final layer. Every neuron in this layer predicts one of the output classes, e.g., the normal class or an arrhythmia class. For an input ECG heartbeat, the neuron that fires more often is considered as the winner, i.e., its corresponding class is considered as the predicted class for the input ECG heartbeat. Unlike STDP, R-STDP follows a supervised learning procedure. During the training stage, if the prediction of a winner neuron k is correct, a reward signal is applied to all its input synapses. In other words, the reward is applied to all synaptic weights ψ^q_{jk}, where j represents all the pre-synaptic neurons from all windows q that are connected to neuron k, i.e., the winner neuron. Similarly, if the prediction is incorrect, a punishment signal is applied. Different learning rules exist for employing R-STDP. Here, we use the following equations which are inspired from the method in [42].

$$\Delta\psi = \begin{cases} a_r^+ \times \psi(\psi_{\max} - \psi) & t_{\text{post}} > t_{\text{pre}} \\ a_r^- \times \psi(\psi_{\max} - \psi) & t_{\text{post}} \leq t_{\text{pre}} \end{cases} \tag{12}$$

$$\Delta\psi = \begin{cases} a_p^- \times \psi(\psi_{\max} - \psi) & t_{\text{post}} > t_{\text{pre}} \\ a_p^+ \times \psi(\psi_{\max} - \psi) & t_{\text{post}} \leq t_{\text{pre}} \end{cases} \tag{13}$$

In the above equations, a_r^+ and a_r^- are learning rates in the reward situation, while a_p^+ and a_p^- are learning rates in the punishment situation. As shown in (12), in the reward situation, i.e., when the prediction is correct, a positive reward proportional to a_r^+ is given to the synapses whose pre-synaptic neurons spike before the winner neuron, i.e., those with $t_{\text{post}} > t_{\text{pre}}$. A negative reward proportional to a_r^- is given to all the other connected synapses, i.e., those with $t_{\text{post}} \leq t_{\text{pre}}$. As shown in (13), the negative and positive signals are reversed in the punishment situation, i.e., when the prediction is incorrect.

The term $\psi(\psi_{\max} - \psi)$ highly increases the probability that the trained weights stay near zero or $\psi_{\max}$. This is because $\Delta\psi$ is zero at $\psi = 0$ and $\psi = \psi_{\max}$. Basically, once a weight is pushed

to either of the two sides, it becomes more difficult to change it. Unlike in pattern extraction in the STDP layer, this is desired in classification in the final layer. This is because we would like a captured pattern from the STDP layer to have either no or full contribution to the prediction of an output class.

G. Training Method

As shown in Fig. 1, there are four trainable layers in the proposed model. Since STDP is unsupervised but R-STDP is supervised, the layers are trained successively. In specific, first the Gaussian layer is trained. Next, the STDP and inhibitory layers are trained simultaneously, and finally, the R-STDP layer is trained.

The initial weights in the STDP layer are set randomly by a normal distribution. The mean should be large enough to let post-synaptic neurons generate enough number of spikes. The learning rate is initially set to a large value in order to allow the weights to change rapidly. This helps the network detect various patterns. To fine-tune the weights, the learning rate is gradually decreased with every epoch. The initial weights of inhibitory and R-STDP layers are also set randomly by a normal distribution.

All weights in the entire network need to be positive, and therefore, negative weights are clipped to zero. However, the backward synapses in the inhibitory layer need to have negative weights, and therefore, only for these synapses, positive weights are clipped to zero.

Note that similar to many other neural network based ECG classification methods, once training of the network is complete, the trained weights are freezed. In other words, during inference (test), the learning rules are not in effect and the weights do not change. Therefore, neurons operate according to equation (1) with constant weights during inference, i.e., during real-time ECG classification.

IV. Experimental Evaluation

A. Setup

The proposed solution is implemented based on Brian2 which is a Python package for evaluation of spiking neural networks [53]. Our source code is available online [54].

The proposed solution is evaluated and compared with previous works using MIT-BIH ECG arrhythmia database. The ECG signals in this database are independently labeled by two or more cardiologists. Two groups of ECG signals called DS1 and DS2 are available in this database. DS1 includes representative samples of different ECG signals and artifacts that an arrhythmia detector might encounter in routine clinical practice. DS2 includes complex arrhythmia patterns [55].

In order to provide thorough and fair comparisons, we employ the exact same data as the previous works. In specific, the following patients from the DS2 group are used as test data: 200, 201, 202, 203, 205, 207, 208, 209, 210, 212, 213, 214, 215, 219, 220, 221, 222, 223, 228, 230, 231, 232, 233 and 234. This is the same set of data employed in [17], [19], [23]. All the details for this dataset, such as patients' age and sex, are available in [56].

TABLE I
Confusion Matrix

	Classification Result			
Ground Truth	NOR	VEB	FUS	Q
NOR	44239	73	99	0
VEB	872	3858	78	0
FUS	191	32	389	0
Q	5	2	1	0

We follow a patient-specific training method similar to [17]–[19], [23], in which the model is trained individually for every patient. The training is not performed in real-time. Once a patient's model is trained, continuous ECG monitoring and heartbeat classification can be performed in real-time on the wearable device based on the trained model of that patient. Training data for every patient is formed based on combining two sets of data, in specific, randomly selected representative heartbeats from all arrhythmia classes in DS1, plus the first five minutes of the patient's own ECG signal. Employing the first five minutes of the patient's own ECG signal is in compliance with AAMI standards [57] and has been used in many studies [17]–[19], [23]. Note that the first five minutes are skipped in the test data.

B. Confusion Matrix

In our experimental evaluation, the proposed model is trained to classify every heartbeat into the following classes. *NOR:* normal beats, *VEB:* ventricular ectopic beats which include premature ventricular contraction and ventricular escape beats, *FUS:* fusion of ventricular and normal beats, and *Q:* paced beats, fusion of paced and normal beats and unclassifiable beats. The resulting confusion matrix is presented in Table I. Comparison with previous works will be presented in Section IV-D, Table II.

C. Real-Time Operation and Energy Consumption

Besides accuracy, two other important factors for continuous monitoring on wearable devices are computational intensity and energy consumption of the ECG classification method. Note that it is only the inference (test) phase which is executed repeatedly on the wearable device and not the training phase. Therefore, it is only the inference which needs to be low power and meet timing requirements for continuous execution. In the following, the timing and energy consumption of the inference phase of the proposed solution are discussed.

Timing: The neurons' operating time step is 1 ms, i.e., the operating frequency is 1 KHz. Every heartbeat signal is provided to the network for 200 ms. At the end of this time duration, the network provides its classification result. This configuration supports classification of up to $60 \times 1000/200 = 300$ beats per minute, and therefore, it is suitable for real-time operation.

Energy Consumption: Once the proposed model is trained, it is simulated according to the method presented in [30], [58] in order to estimate its energy consumption in classifying the ECG signals. The input heartbeat signals are applied to the network and the neurons are allowed to fire. For every heartbeat, we fully simulate the entire network, i.e., all neurons and all synapses from all layers, for a period of 200 ms. During this period,

TABLE II
COMPARING THE PROPOSED SOLUTION WITH PREVIOUS WORKS IN TERMS OF CLASSIFICATION PERFORMANCE, NETWORK TYPE, REAL-TIME OPERATION, AND ENERGY CONSUMPTION. TABLE ENTRIES ARE SORTED BASED ON THEIR ENERGY CONSUMPTION

	Acc	Sen	Spe	Ppr	F_1	Type	Energy per heartbeat (Joules)	Real-time?
Hu *et al.* [18]	94.8	78.9	96.8	75.8	77.3	MLP	–	–
Crippa *et al.* [15]	98.9	95.7	99.1	87.3	91.3	KLT, GMM	–	–
Rajpurkar *et al.* [20]	–	82.7	–	80.9	–	CNN	1643 J	No
Zhang *et al.* [21]	–	93.6	–	98.4	95.9	CNN	38.2 J	No
Kachuee *et al.* [22]	93.4	–	–	–	–	CNN	1.17 J	Near Real-time
Kiranyaz *et al.* [17]	98.6	95.0	98.1	89.5	92.2	FFT, CNN	37 mJ	Yes
Saadatnejad *et al.* [23]	99.2	93.0	99.8	98.2	95.5	Wavelet, RNN	35 mJ	Yes
Ince *et al.* [19]	97.6	83.4	98.1	87.4	85.4	Wavelet, MLP	4 mJ	Yes
Kolagasioglu *et al.* [35]	95.5	–	–	–	–	Wavelet, SNN	614 μJ	Yes
Lee *et al.* [13]	97.2	–	–	–	–	Wavelet	5.97 μJ	Yes
Proposed	97.9	80.2	99.8	97.3	88.0	SNN	1.78 μJ	Yes

all neurons may receive spikes from their input synapses, fire at any time, and send spikes to their output synapses. When a neuron fires a spike, we consider that 50 pJ (pico Joules) of energy is consumed [30]. The connected output synapses transfer this spike to other neurons. For every spike transfer, i.e., every synaptic event, 147 pJ of energy is counted [30]. The simulation results show that the energy consumption for classification of every heartbeat is 1.78 μJ (micro Joules).

D. Comparison With Previous Works

Table II compares the proposed solution with previous works, in terms of classification performance, network type, real-time operation, and energy consumption. Four statistical metrics, namely accuracy, sensitivity (recall), specificity, and positive predictivity (precision) are extracted from the confusion matrix for binary classification of VEB beats from non-VEB beats. Since there exists a trade-off between sensitivity and positive predictivity, they are combined into another metric called F_1 score as the following.

$$F_1 = \frac{2}{\frac{1}{\text{Sen}} + \frac{1}{\text{Ppr}}} \tag{14}$$

Hu *et al.* [18] proposed a mixture of experts algorithm based on multi-level perceptron (MLP). Our proposed solution has higher classification performance compared to this method. Accuracy and F_1 score are 3.1% and 10.7% higher, respectively. Crippa *et al.* [15] proposed a Gaussian mixture model of Karhunen-Love transform. Compared to this algorithm, our proposed method has 1% and 3.3% lower accuracy and F_1 score, respectively. Enough details are not available to estimate the energy consumption of the above methods.

Details of the ECG classification algorithms in [17], [19]–[23] are available. Since they are software-based solutions, we have implemented them on ARM Cortex A53 which is a low-power processor and measured their execution times. Energy consumption is then estimated as the nominal power rating of the processor (0.9 Watts) multiplied by the average execution time in classifying different heartbeats. The 34-layer CNN in [20] takes about 1825 seconds to classify every heartbeat, and consumes about 1643 Joules of energy. This compute-intensive method does not meet timing requirements for continuous real-time execution. The CNN model in [21] has large number of filters in its convolution layers. This method takes about 42.5 seconds to classify every heartbeat, and consumes about 38.2 Joules of energy. The CNN model in [22] is relatively smaller. It has 11 convolution layers. It takes about 1.3 seconds to classify every heartbeat, and consumes about 1.17 J of energy. Kiranyaz *et al.* proposed a lightweight algorithm for real-time ECG classification based on fast Fourier transform (FFT) and one-dimensional convolution layers [17]. It takes about 41 ms to classify every heartbeat, and consumes about 37 mJ (mili Joules). Another lightweight algorithm is proposed in [23] which is based on wavelet transform and small recurrent neural networks (RNN). This method takes about 39 ms to classify every heartbeat, and consumes about 35 mJ. Ince *et al.* [19] proposed to select the configuration of artificial neural networks using particle swarm optimization. The input to the selected neural network is provided by wavelet transform followed by PCA. It takes about 4.5 ms to classify every heartbeat, and consumes about 4 mJ.

The proposed algorithm has comparable classification performance with respect to the above solutions. Accuracy is 1.3% lower than [23] and 4.5% higher than [22]. F_1 score is 7.5% lower than [23] and 2.6% higher than [19]. More importantly, energy consumption of the proposed solution is about 9 orders of magnitude (billion times) smaller than the CNN-based algorithm in [20], about 6 orders of magnitude smaller than [22], and even about 3 to 4 orders of magnitude smaller than the real-time and lightweight methods in [17], [23], and [19].

The SNN-based ECG classification method in [35] consists of two parts. First, wavelet transform is employed for feature extraction, and a correlation matrix for feature selection. Next, based on the selected feature set, a liquid state machine (LSM) which is a specific type of spiking neural network is employed to cluster the input signal. As the author stated (page 41), the second part consumes 111.62 nW, and the first part is very similar to the circuit in [2]. Therefore, as reported in [2], the energy consumption of the first part is 614 μJ. Hence, total energy consumption is 614.111 μJ. Compared to [35], the proposed method achieves higher accuracy, and its energy consumption is 345 X smaller because it entirely operates in the spike domain. Other SNN-based solutions that process ECG signals include extraction of the heart rate [36], applying a band-pass filter to the input ECG signal [37], and detecting perturbations in the signal [38].

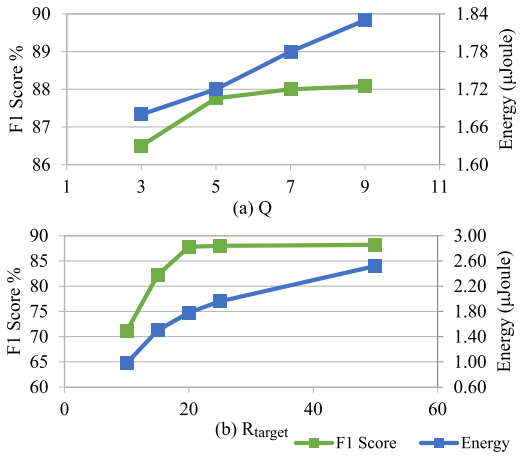

Fig. 7. F_1 score and energy consumption versus (a) parameter Q in segmentation, and (b) R_{target} in Gaussian layer.

Lee *et al.* [13] proposed a low-power wireless ECG acquisition and classification system. The ECG classification part is based on Haar wavelet. The extracted wavelet features are compared against eight previously-selected feature vectors in order to classify the ECG signal. Hence, the classification algorithm is relatively lightweight and does not require training. The energy consumption for ECG classification part is reported as 5.97 μJ.

As summarized in Table II, with comparable classification performance, the proposed method consumes significantly smaller energy than previous methods in classification of heartbeats.

E. Efficiency of Segmentation

This and the following sections provide more insight into the proposed SNN-based solution. This section focuses on segmentation.

Since the windows are overlapped, the larger the number of windows, i.e., Q, the more the amount of redundant computations, and hence, the higher the energy consumption. At the other hand, more number of windows help to increase accuracy by better analyzing different parts of the input ECG signal.

The above trade-off is evaluated experimentally and the result is shown in Fig. 7(a). The figure shows how the energy consumption and F_1 score vary with Q. F_1 score is almost saturated beyond $Q = 7$, but energy consumption is increased. Therefore, this point is selected as the value of parameter Q.

F. Efficiency of Gaussian Layer

The Gaussian layer helps to reduce the energy consumption. The smaller the value of R_{target} in the Gaussian layer, the smaller the number of fired spikes in the entire network, and thus, the lower the energy consumption. See Section III-C. At the other hand, the larger the value of R_{target}, the higher the classification performance. This is because more spikes help with more accurate pattern extraction.

The above trade-off is evaluated experimentally. Fig. 7(b) shows how the energy consumption and F_1 score vary with R_{target}. F_1 score is almost saturated beyond $R_{target} = 20$, but energy consumption is increased. Hence, this point is selected as the value of R_{target}.

Without the Gaussian layer, the encoder cells would be directly connected to the neurons in the STDP layer, the number of spikes would not be adjusted, and the resulting energy consumption would be 4.27 μJ, i.e., 2.4 X higher.

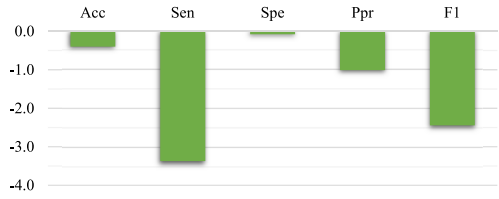

Fig. 8. Degradation of classification performance metrics (in percent) if STDP learning rule is not optimized.

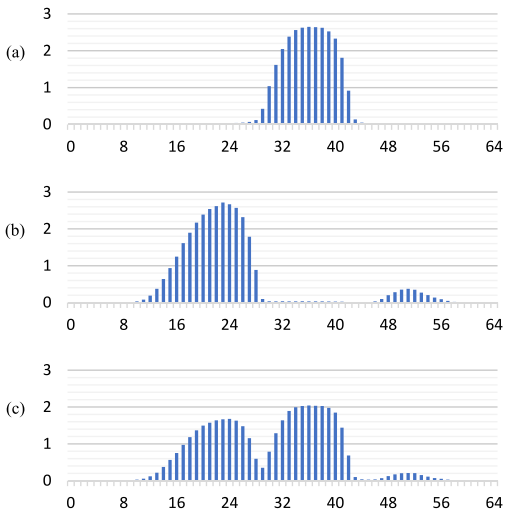

Fig. 9. Trained input weights in the STDP layer in window 4, for neurons (a) N_0, (b) N_1, and (c) N_2.

G. Efficiency of STDP Pattern Extraction

Fig. 8 shows degradation of classification performance metrics if Equation (7) would have been employed instead of Equation (8), i.e., the optimized STDP learning rule. Sensitivity and positive predictivity would have been 3.4% and 1.0% lower, respectively. F_1 score would have been 2.5% lower. Therefore, the optimized STDP learning rule improves classification performance.

To provide more insight into the inner workings of STDP-based pattern extraction, Fig. 9 shows the trained input synaptic weights for three sample neurons in the STDP layer, namely, neurons N_0, N_1 and N_2 from window 4. As illustrated in the figure, the trained weights are non-binary and vary based on the captured patterns. This shows that the proposed STDP learning rule in (8) is effective in training the weights as desired. The figure also shows that the three captured patterns are different, which means the proposed inhibitory learning rule in (11) has correctly trained the negative weights, and thus, the inhibitory neurons have succeeded in diverging the attention of the STDP neurons.

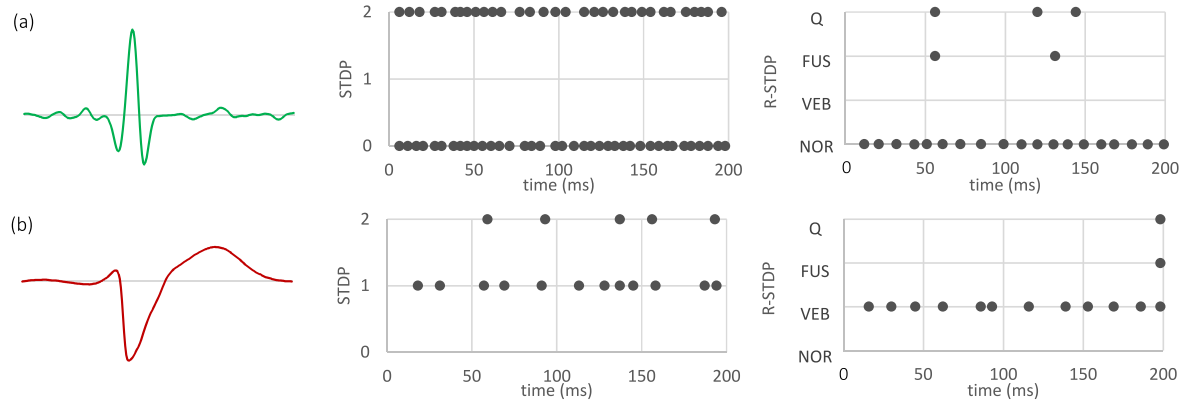

Fig. 10. Fired spikes for two sample heartbeat signals: (a) normal beat, (b) premature ventricular contraction beat.

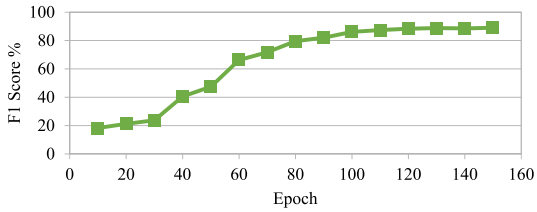

Fig. 11. F_1 score versus the number of training epochs.

In order to study the three sample neurons in action, Fig. 10(a) shows their fired spikes for a normal input heartbeat. Neurons N_0 and N_2 fire many spikes. In the next layer, i.e., R-STDP, there are four output neurons that fire 19, 0, 2, and 3 spikes. Since the first output neuron fires more spikes than the others, it determines the predicted class, i.e., NOR, which is correct. Fig. 10(b) shows the fired spikes for a premature ventricular contraction beat. Neuron N_1 fires many spikes and neuron N_2 fires a few spikes. The four output neurons in the R-STDP layer fire 0, 12, 1, and 1 spikes. Hence, the second output neuron determines the predicted class, i.e., VEB, which is also correct.

We can conclude that neuron N_0 is trained to be sensitive to a pattern in the NOR class, neuron N_1 is trained to be sensitive to a pattern in the VEB class, and neuron N_2 is sensitive to both. Hence, neuron N_2 does not help much with the classification. Investigating the weights of output synapses of neuron N_2, i.e., its R-STDP weights, reveal that they are trained as small values. This shows that the R-STDP weights are trained as desired.

Fig. 11 shows how increasing the number of training epochs improves classification performance, in specific, F_1 score. Similar to other neural network based solutions, enough number of training epochs are required to achieve an acceptable performance but eventually a saturation point is reached.

V. Conclusion

This manuscript proposed an ECG classification algorithm in which both the pattern extraction and the classification steps are implemented based on spiking neural networks. In addition, the learning rules are optimized to better fit the ECG classification domain. As a result, the achieved accuracy is comparable to previous methods while the energy consumption is multiple orders of magnitude smaller than previous neural network based ECG classification methods. Hence, the proposed algorithm is suitable to be included as part of ultra low-power wearable cardiac monitoring devices.

References

[1] D. Sopic *et al.*, "Real-time event-driven classification technique for early detection and prevention of myocardial infarction on wearable systems," *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 5, pp. 982–992, Oct. 2018.
[2] N. Bayasi *et al.*, "A 65-nm low power ECG feature extraction system," in *Proc. Int. Symp. Circuits Syst.*, 2015, pp. 746–749.
[3] D. D. He and C. G. Sodini, "A 58 nw ECG ASIC with motion-tolerant heartbeat timing extraction for wearable cardiovascular monitoring," *IEEE Trans. Biomed. Circuits Syst.*, vol. 9, no. 3, pp. 370–376, Jun. 2015.
[4] X. Tang, Q. Hu, and W. Tang, "A real-time QRS detection system with PR/RT interval and ST segment measurements for wearable ECG sensors using parallel delta modulators," *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 4, pp. 751–761, Aug. 2018.
[5] J. M. Bote, J. Recas, F. Rincón, D. Atienza, and R. Hermida, "A modular low-complexity ECG delineation algorithm for real-time embedded systems," *IEEE J. Biomed. Health Inform.*, vol. 22, no. 2, pp. 429–441, Mar. 2018.
[6] X. Wang, Q. Gui, B. Liu, Z. Jin, and Y. Chen, "Enabling smart personalized healthcare: A hybrid mobile-cloud approach for ECG telemonitoring," *IEEE J. Biomed. Health Inform.*, vol. 18, no. 3, pp. 739–745, May 2014.
[7] C. Chou, E. Chang, H. Li, and A. Wu, "Low-complexity privacy-preserving compressive analysis using subspace-based dictionary for ECG telemonitoring system," *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 4, pp. 801–811, Aug. 2018.
[8] S. Song *et al.*, "A 769 µW battery-powered single-chip SoC with BLE for multi-modal vital sign monitoring health patches," *IEEE Trans. Biomed. Circuits Syst.*, 2019.
[9] G. Biagetti, P. Crippa, L. Falaschetti, S. Orcioni, and C. Turchetti, "Wireless surface electromyograph and electrocardiograph system on 802.15.4," *IEEE Trans. Consum. Electron.*, vol. 62, no. 3, pp. 258–266, Aug. 2016.
[10] P. Li *et al.*, "High-performance personalized heartbeat classification model for long-term ECG signal," *IEEE Trans. Biomed. Eng.*, vol. 64, no. 1, pp. 78–86, Jan. 2017.
[11] T. Teijeiro, P. Félix, J. Presedo, and D. Castro, "Heartbeat classification using abstract features from the abductive interpretation of the ECG," *IEEE J. Biomed. Health Inform.*, vol. 22, no. 2, pp. 409–420, Mar. 2018.
[12] L.-Y. Shyu, Y.-H. Wu, and W. Hu, "Using wavelet transform and fuzzy neural network for VPC detection from the Holter ECG," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 7, pp. 1269–1273, Jul. 2004.
[13] S. Y. Lee *et al.*, "Low-power wireless ECG acquisition and classification system for body sensor networks," *IEEE J. Biomed. Health Inform.*, vol. 19, no. 1, pp. 236–246, Jan. 2015.
[14] S. K. Jain and B. Bhaumik, "An energy efficient ECG signal processor detecting cardiovascular diseases on smartphone," *IEEE Trans. Biomed. Circuits Syst.*, vol. 11, no. 2, pp. 314–323, Apr. 2017.

[15] P. Crippa *et al.*, "Multi-class ECG beat classification based on a Gaussian mixture model of Karhunen-Loève transform," *Int. J. Simul.: Syst., Sci., Technol.*, vol. 16, no. 1, 2015.
[16] R. Hoekema, G. J. H. Uijen, and A. Van Oosterom, "Geometrical aspects of the interindividual variability of multilead ECG recordings," *IEEE Trans. Biomed. Eng.*, vol. 48, no. 5, pp. 551–559, May 2001.
[17] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ECG classification by 1-D convolutional neural networks," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 3, pp. 664–675, Mar. 2016.
[18] Y. H. Hu, S. Palreddy, and W. J. Tompkins, "A patient-adaptable ECG beat classifier using a mixture of experts approach," *IEEE Trans. Biomed. Eng.*, vol. 44, no. 9, pp. 891–900, Sep. 1997.
[19] T. Ince, S. Kiranyaz, and M. Gabbouj,, "A generic and robust system for automated patient-specific classification of electrocardiogram signals," *IEEE Trans. Biomed. Eng.*, vol. 56, no. 5, pp. 1415–1426, May 2009.
[20] P. Rajpurkar, A. Y. Hannun, M. Haghpanahi, C. Bourn, and A. Y. Ng, "Cardiologist-level arrhythmia detection with convolutional neural networks," 2017, *arXiv:1707.01836*.
[21] W. Zhang, L. Yu, L. Ye, W. Zhuang, and F. Ma, "ECG signal classification with deep learning for heart disease identification," in *Proc. Int. Conf. Big Data Artif. Intell.*, 2018, pp. 47–51.
[22] M. Kachuee, S. Fazeli, and M. Sarrafzadeh, "ECG heartbeat classification: A deep transferable representation," in *Proc. IEEE Int. Conf. Healthcare Inform.*, 2018, pp. 443–444.
[23] S. Saadatnejad, M. Oveisi, and M. Hashemi, "LSTM-based ECG classification for continuous monitoring on personal wearable devices," *IEEE J. Biomed. Health Inform.*, 2019.
[24] M. H. Foroozannejad, M. Hashemi, A. Mahini, B. M. Baas, and S. Ghiasi, "Time-scalable mapping for circuit-switched GALS chip multiprocessor platforms," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 33, no. 5, pp. 752–762, May 2014.
[25] M. Hashemi *et al.*, "Throughput-memory footprint trade-off in synthesis of streaming software on embedded multiprocessors," *ACM Trans. Embedded Comput. Syst.*, vol. 13, no. 3, 2013, Art. no. 46.
[26] M. H. Foroozannejad *et al.*, "Postscheduling buffer management trade-offs in streaming software synthesis," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 17, no. 3, 2012, Art. no. 27.
[27] K. M. Barijough *et al.*, "Implementation-aware model analysis: The case of buffer-throughput tradeoff in streaming applications," in *Proc. ACM SIGPLAN Notices*, vol. 50, no. 5, 2015, Art. no. 11.
[28] I. Yeo, M. Chu, and B. G. Lee, "A power and area efficient CMOS stochastic neuron for neural networks employing resistive crossbar array," *IEEE Trans. Biomed. Circuits Syst.*, 2019.
[29] N. Qiao *et al.*, "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128 K synapses," *Frontiers Neurosci.*, vol. 9, 2015, Art. no. 141.
[30] N. Qiao and G. Indiveri, "Scaling mixed-signal neuromorphic processors to 28 nm FD-SOI technologies," in *Proc. IEEE Biomed. Circuits Syst. Conf.*, 2016, pp. 552–555.
[31] N. Qiao and G. Indiveri, "Analog circuits for mixed-signal neuromorphic computing architectures in 28 nm FD-SOI technology," in *Proc. IEEE SOI-3D-Subthreshold Microelectronics Technol. Unified Conf. (S3S)*, 2017, pp. 1–4.
[32] S. Moradi and G. Indiveri, "An event-based neural network architecture with an asynchronous programmable synaptic memory," *IEEE Trans. Biomed. Circuits Syst.*, vol. 8, no. 1, pp. 98–107, 2013.
[33] S. Moradi, S. A. Bhave, and R. Manohar, "Energy-efficient hybrid CMOS-NEMS LIF neuron circuit in 28 nm CMOS process," in *Proc. Symp. Ser. Comput. Intell.*, 2017, pp. 1–5.
[34] G. Indiveri, F. Corradi, and N. Qiao, "Neuromorphic architectures for spiking deep neural networks," *Tech. Digest—Int. Electron Devices Meeting*, vol. 2016, pp. 4.2.1–4.2.4, 2015.
[35] E. Kolagasioglu and A. Zjajo, "Energy efficient feature extraction for single-lead ECG classification based on spiking neural networks," Ph.D. dissertation, Dept. Elect. Eng., Delft Univ. Technol., Delft, The Netherlands, 2018.
[36] A. Das *et al.*, "Unsupervised heart-rate estimation in wearables with liquid states and a probabilistic readout," *Neural Netw.*, vol. 99, pp. 134–147, 2018.
[37] Q. Ma *et al.*, "A low-power neuromorphic bandpass filter for biosignal processing," in *Proc. IEEE Wireless Microw. Technol. Conf.*, 2013, pp. 1–3.
[38] R. Cattaneo, "ECG signals classification using neuromorphic hardware," Ph.D. dissertation, Dept. Electron. Eng., ETH Zurich, Zurich, Switzerland, 2018.
[39] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers Comput. Neurosci.*, vol. 9, no. August, pp. 1–9, 2015.
[40] T. Masquelier and S. J. Thorpe, "Unsupervised learning of visual features through spike timing dependent plasticity," *PLoS Comput. Biol.*, vol. 3, no. 2, pp. 247–257, 2007.
[41] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "STDP-based spiking deep convolutional neural networks for object recognition," *Neural Netw.*, vol. 99, pp. 56–67, 2018.
[42] M. Mozafari *et al.*, "First-spike-based visual categorization using reward-modulated STDP," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6178–6190, Dec. 2018.
[43] Z. Bing *et al.*, "A survey of robotics control based on learning-inspired spiking neural networks," *Frontiers Neurorobotics*, vol. 12, 2018, Art. no. 35.
[44] N. Kasabov and E. Capecci, "Spiking neural network methodology for modelling, classification and understanding of EEG spatio-temporal data measuring cognitive processes," *Inf. Sci.*, vol. 294, pp. 565–575, 2015.
[45] W. Gerstner, *et al.*, *Neuronal Dynamics, from Single Neurons to Networks and Models of Cognition*. Cambridge, U.K.: Cambridge Univ. Press, 2014.
[46] P. Dayan and L. F. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. Cambridge, MA, USA: MIT Press, 2001, ch. 1, pp. 1–43.
[47] M. Meister and M. J. I. Berry, "The neural code of the retina," *Neuron*, vol. 22, no. 3, pp. 435–450, 1999.
[48] S. Song, K. D. Miller, and L. F. Abbott, "Competitive Hebbian learning through spike-timing-dependent synaptic plasticity," *Nature Neurosci.*, vol. 3, no. 9, 2000, Art. no. 919.
[49] C. C. Garner *et al.*, "Inhibitory plasticity balances excitation and inhibition in sensory," no. December, 2012.
[50] N. Srinivasa and Q. Jiang, "Stable learning of functional maps in self-organizing spiking neural networks with continuous synaptic plasticity," *Frontiers Comput. Neuro.*, vol. 7, 2013, Art. no. 10.
[51] N. Srivastava *et al.*, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[52] N. Frémaux and W. Gerstner, "Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules," *Frontiers Neural Circuits*, vol. 9, no. January, 2016, Art. no. 85.
[53] M. Stimberg *et al.*, "Brian 2 documentation." [Online]. Available: http://brian2.readthedocs.io
[54] Source code. [Online]. Available: http://lis.ee.sharif.edu
[55] R. G. Mark and G. B. Moody, "MIT-BIH arrhythmia database," 1997. [Online]. Available: http://ecg.mit.edu/dbinfo.html
[56] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH arrhythmia database," *IEEE Eng. Med. Biol. Mag.*, vol. 20, no. 3, pp. 45–50, May/Jun. 2001.
[57] *AAMI-Recommended Practice: Testing and Reporting Performance Results of Ventricular Arrhythmia Detection Algorithms*. Arlington, VA, USA: Assoc. Advancement Med. Instrum., 1987.
[58] Y. Cao, Y. Chen, and D. Khosla, "Spiking deep convolutional neural networks for energy-efficient object recognition," *Int. J. Comput. Vision*, vol. 113, no. 1, pp. 54–66, 2015.

Alireza Amirshahi received the B.Sc. and M.Sc. degrees in electrical engineering from the Sharif University of Technology, Tehran, Iran, in 2017 and 2019, respectively. His research interests include machine learning for biomedical signal processing.

Matin Hashemi received the B.Sc. degree in electrical engineering from the Sharif University of Technology, Tehran, Iran, in 2005, and the M.Sc. and Ph.D. degrees in computer engineering from the University of California, Davis, CA, USA, in 2008 and 2011, respectively. He is currently an Assistant Professor in electrical engineering with the Sharif University of Technology. His research interests include algorithm design and hardware acceleration for machine learning, signal processing, and big data applications.