Real-Time Patient-Specific ECG Classification by 1D Convolutional Neural Networks

Article *in* IEEE transactions on bio-medical engineering · August 2015

DOI: 10.1109/TBME.2015.2468589 · Source: PubMed

3 authors:

Serkan Kiranyaz
Qatar University
380 PUBLICATIONS **18,134** CITATIONS

SEE PROFILE

Turker Ince
GIU Berlin
134 PUBLICATIONS **9,662** CITATIONS

SEE PROFILE

Moncef Gabbouj
Tampere University
1,029 PUBLICATIONS **28,384** CITATIONS

SEE PROFILE

Real-Time Patient-Specific ECG Classification by 1-D Convolutional Neural Networks

Serkan Kiranyaz*, Turker Ince, and Moncef Gabbouj, *Fellow, IEEE*

***Abstract*—*Goal:* This paper presents a fast and accurate patient-specific electrocardiogram (ECG) classification and monitoring system. *Methods:* An adaptive implementation of 1-D convolutional neural networks (CNNs) is inherently used to fuse the two major blocks of the ECG classification into a single learning body: feature extraction and classification. Therefore, for each patient, an individual and simple CNN will be trained by using relatively small common and patient-specific training data, and thus, such patient-specific feature extraction ability can further improve the classification performance. Since this also negates the necessity to extract hand-crafted manual features, once a dedicated CNN is trained for a particular patient, it can solely be used to classify possibly long ECG data stream in a fast and accurate manner or alternatively, such a solution can conveniently be used for real-time ECG monitoring and early alert system on a light-weight wearable device. *Results:* The results over the MIT-BIH arrhythmia benchmark database demonstrate that the proposed solution achieves a superior classification performance than most of the state-of-the-art methods for the detection of ventricular ectopic beats and supraventricular ectopic beats. *Conclusion:* Besides the speed and computational efficiency achieved, once a dedicated CNN is trained for an individual patient, it can solely be used to classify his/her long ECG records such as Holter registers in a fast and accurate manner. *Significance:* Due to its simple and parameter invariant nature, the proposed system is highly generic, and, thus, applicable to any ECG dataset.**

***Index Terms*—Convolutional neural networks (CNNs), patient-specific ECG classification, real-time heart monitoring.**

I. Introduction

DESPITE the easiness of acquiring the data, there are still challenges ahead of us in order to extract reliable information from biomedical signals. Each heartbeat in the cardiac cycle shows the time evolution of the heart's electrical activity, which is made up of distinct electrical depolarization-repolarization patterns of the heart. For an expert cardiologist, any anomaly over the heart rate or rhythm or change in the morphological pattern over a recorded ECG waveform can easily be detected as an indication of an arrhythmia. However, this can turn out to be very challenging task for an automatic computerized system due to several reasons. Certain contaminations of biomedical signals to physiological artefact and external noise as well as imbalanced classes among biomedical signals (e.g., N- and S-type beats in an ECG signal) make the system's performance and accuracy significantly varying from patient to patient. Particularly, the time-varying dynamics and the morphological characteristics of ECG signals show significant variations for different patients and under different temporal and physical conditions. Even for the ECG of a healthy subject, which appears to be deterministic, the shapes of QRS complex, P waves, and R–R intervals will not be the same from one beat to the other under different circumstances [1].

There have been several methods for generic and fully automatic ECG classification based on signal processing techniques, such as frequency analysis [2], wavelet transform [3] and filter banks [4], statistical [5] and heuristic approaches [6], hidden Markov models [7], support vector machines [8], artificial neural networks (ANNs) [9], and mixture-of-experts method [10]. Generally speaking, they have not performed well in practice due to the aforementioned interpatient variations of the ECG signals, and thus, they usually exhibit a common drawback of having an inconsistent performance when, for instance, classifying a new patient's ECG signal. This makes them unreliable to be widely used clinically or in practice, and they tend to have high variations in their accuracy and efficiency for larger databases [11], [12].

Another severe problem is the lack of application of the common practice when evaluating and testing a particular method over a benchmark dataset. For this purpose, the *Association for the Advancement of Medical Instrumentation* (AAMI) provides standards and recommended practices for performance results of automated arrhythmia detection algorithms [13]. However, among many methods in the literature, only few [10], [14]–[18] have in fact used the AAMI standards along with the complete data from the benchmark MIT-BIH arrhythmia database [22]. Among all, only few of them with a patient-specific design [10], [12], [15]–[18] have, in particular, demonstrated significant performance improvements over the automatic and generic ECG classification methods thanks to their ability to adapt or optimize the classifier body according to each patient's ECG signal.

The aforementioned patient-specific ECG classification systems have a common approach with two major operations: feature extraction and training classification over the extracted features. They demonstrated that the ECG classification performance strongly depends on the characterization power of the features extracted from the ECG data. In the ECG classification literature, a vast number of features, their combinations, and feature selection approaches have been proposed [20]. In a former work, *Hermite* transform coefficients [15] achieved such a performance that is significantly higher than the others.

Manuscript received March 13, 2015; revised June 7, 2015, June 22, 2015, and July 22, 2015; accepted August 7, 2015. Date of publication; date of current version. *Asterisk indicates corresponding author.*

*S. Kiranyaz is with Electrical Engineering Department, College of Engineering, Qatar University, Doha, Qatar (e-mail: mkiranyaz@qu.edu.qa).

T. Ince is with the Izmir University of Economics

M. Gabbouj is with the Tampere University of Technology

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

Digital Object Identifier 10.1109/TBME.2015.2468589

Due to its time–frequency localization properties, the wavelet transform proves to be an efficient tool for analyzing nonstationary ECG signals [19]. In our prior work, Kiranyaz *et al.* [16], [17] that achieved superior performance than [15], we used translation-invariant dyadic wavelet transform to extract morphological features, and in order to avoid the well-known "Curse of Dimensionality" phenomenon and to significantly reduce redundancies in such a high-dimensional data space, the dimension of the input feature vectors has further been reduced by using principal component analysis (PCA). The lower dimensional morphological feature vector was then combined with two critical temporal features to form the final feature vector. However, using such fixed and hand-crafted features may not represent the characteristics of the underlying signal in an optimal way and obviously this is against the philosophy of a "patient-specific" approach since the same set of features will be used for all patients under all circumstances. The true "patient-specific" solution indeed requires the design of the best possible features for each individual ECG data. Moreover, extracting several features, especially in the transform domains along with the postprocessing methods such as PCA may significantly increase the computational complexity of the overall process, and this may hinder them from the usage in light-weight applications (e.g., mobile or wearable health monitoring devices) or for the classification of large ECG records such as Holter registers.

In order to address such deficiencies and drawbacks, in this paper, we propose a novel ECG classification approach based on adaptive 1-D convolutional neural networks (CNNs). CNNs are hierarchical neural networks whose convolutional layers alternate with subsampling layers, reminiscent of simple and complex cells in the human visual cortex [21], following with a fully connected layers, which are identical to multilayer perceptrons (MLP). They primarily mimic the human visual system, which can efficiently recognize the patterns and structures (e.g., objects) in a visual scenery. CNNs are now commonly used for the "deep learning" tasks, such as object recognition in large image achieves while achieving the state-of-the-art performances [24]–[26]. To our knowledge, this is the first study, where they are used over 1-D signals, in particular, for the purpose of ECG classification and anomaly detection. With the proposed adaptation over the traditional CNNs, the proposed approach can classify each heart beat with any sampling rate; therefore, voiding the need for any manual feature extraction and postprocessing. With the proper training, the convolutional layers of CNNs can learn to extract patient-specific features, while the MLP layers perform the classification task to produce the final class vectors of each beat. With the limited training data as proposed in [10] and [14]–[17], we shall demonstrate that simple CNNs will suffice to achieve a superior classification performance rather than the complex ones that are commonly used for deep learning tasks. As a result, simple 1-D CNNs are easier to train with only few dozens of back-propagation (BP) epochs, and can, thus, perform the classification task with utmost speed (requiring only few hundreds of 1-D convolutions). This makes them a perfect choice for real-time ECG monitoring and early alert system on light-weight devices. An illustration of the proposed approach is shown in Fig. 1. Finally, we aim to achieve a high level of *robustness* with respect to the variations of the dataset, since the proposed system is designed with a minimum set of parameters and manual settings thanks to the combined learner for feature extraction and classification.

The rest of this paper is organized as follows. Section II outlines the ECG dataset used in this study, and provides a detailed description of the possible raw data representations for the proposed patient-specific heartbeat classification system. The adaptive 1-D CNNs along with the BP training method are presented in Section III. In Section IV, the performance and robustness of the proposed approach are evaluated over the MIT/BIH arrhythmia database using the standard performance metrics, and the results are compared with the previous state-of-the-art works. Finally, Section V concludes this paper.

II. ECG Data Processing

In this study, ECG datasets from the MIT/BIH arrhythmia database [22] are used for the performance evaluation of the proposed patient-specific ECG approach. This benchmark database contains 48 records, each containing two-channel ECG signals for 30-min duration selected from 24-h recordings of 47 individuals. Continuous ECG signals are bandpass filtered at 0.1–100 Hz and then digitized at 360 Hz. The database contains annotation for both timing information and beat class information verified by independent experts. In this study, we followed the identical data partitioning as in [16] and [17] so as to comply with the AAMI ECAR-1987 recommended practice [13]. We used 44 records from the MIT/BIH arrhythmia database, excluding four records, which contain paced heartbeats. The first 20 records (numbered in the range of 100–124), which include representative samples of routine clinical recordings, are used to select representative beats to be included in the common training data. The remaining 24 records (numbered in the range of 200–234) contain uncommon but clinically significant arrhythmias, such as ventricular, junctional, and supraventricular arrhythmias [27]. A total of 83648 beats from all 44 records are used as test patterns for performance evaluation. AAMI recommends that each ECG beat be classified into the following five heartbeat types: N (beats originating in the sinus mode), S (supraventricular ectopic beats), V (ventricular ectopic beats), F (fusion beats), and Q (unclassifiable beats). For all records, we used the modified-lead II signals, and utilized the labels to locate beats in ECG data. The beat detection process is beyond the scope of this paper, as many highly accurate ($> 99\%$) beat detection algorithms have been reported in the literature [19], [23].

The raw data of each beat are represented by 64 or 128 samples by downsampling where the latter is intended for the evaluation of the higher resolution data representation. As illustrated in Fig. 1, in order to learn the morphological structure of the beat, equal number of samples from each side from the R (center) point of the beat are fed into a neuron of the CNN's input layer. In order to learn the temporal characteristics of each beat, a beat trio is formed from its neighbor beats, and is fed into another neuron at the input layer. Therefore, the difference in timing information of the center beat together with its

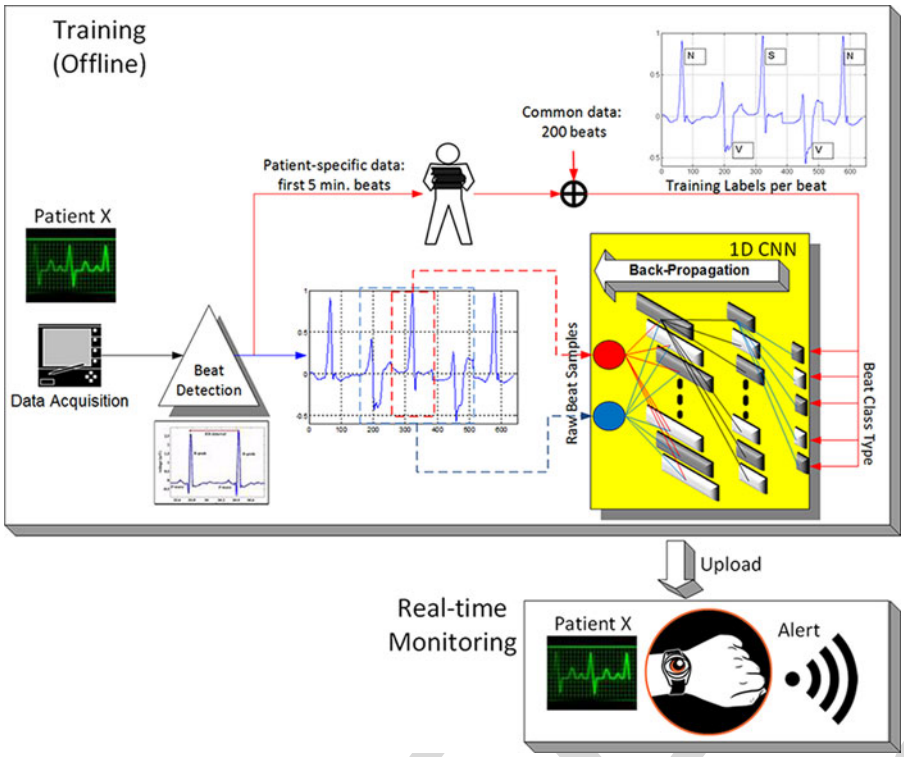

Fig. 1. Overview of the proposed approach in training (offline) and real-time classification and monitoring phases.

neighbors in the beat-trio formation can indicate timing information related ECG anomalies such as the presence of an APC (S) beat. This is the *base* representation of each beat's raw data and on top of this, FFT of each beat (both magnitude and phase) will also be considered as the *extended* raw data representation in the frequency domain. The purpose is to evaluate the performance gain—if any—obtained by such extension in raw-data representation.

The data used for training the individual patient's classifier consist of two parts: global (common to each patient) and local (patient-specific) training patterns. While patient-specific data contain the first 5-min segment of each patient's ECG record and is used as a part of the training data to perform patient adaptation, the global dataset contains a relatively small number of representative beats from each class in the training files, and helps the classifier learn other arrhythmia patterns that are not included in the patient-specific data. This practice conforms to the AAMI recommended procedure allowing the usage of at most 5-min section from the beginning of each patient's recording for training [13].

III. Adaptive 1-D CNNs

As mentioned earlier, adaptive 1-D CNNs are used for both feature extraction and classification of the raw ECG data from each individual patient in the database. In Appendix A, we introduced an overview of the traditional CNNs developed for a 2-D image classification. Accordingly, we shall present the design of our adaptive CNNs in accordance with the traditional CNNs in 2-D and formulate its BP training. Finally, we shall highlight the changes and modifications needed for 1-D CNNs from their 2-D counterparts along with the BP formulations.

To simplify the CNN analogy and to have the freedom of any input layer dimension independent from the CNN parameters, the neurons of the hidden CNN layers are extended such that they are capable of both convolution and downsampling as shown in Fig. 2. This implementation also allows the ability of a "CNN-only" design without the MLP layers. For the illustration purpose, we assume 3×3 kernels ($Kx = Ky = 3$) for all CNN layers in the figure; however, different kernel sizes can also be assigned if desired. The final output of the kth neuron at layer l, s_k^l, is, therefore, the subsampled version of the intermediate output y_k^l. During the forward propagation (FP), the input map of the next layer neuron will be obtained by the cumulation of the final output maps of the previous layer neurons convolved with their individual kernels as follows:

$$x_k^l = b_k^l + \sum_{i=1}^{N_{l-1}} \text{conv2D}(w_{ik}^{l-1}, s_i^{l-1}) \quad (1)$$

where conv2D (.,.) is a regular 2-D convolution without zero padding on the boundaries, x_k^l is the input, b_k^l is the bias of the kth neuron at layer l, and s_i^{l-1} is the output of the ith neuron at layer l–1. w_{ik}^{l-1} is the kernel (weight) from the ith neuron at layer l–1 to the kth neuron at layer l. To accomplish BP training, there are three more elements that are stored for each neuron: the delta error Δ_k^l, downsampled delta error Δ_{sk}^l, and, finally, the derivative of the intermediate output $f'(x_k^l)$, all of which will be explained in the next section.

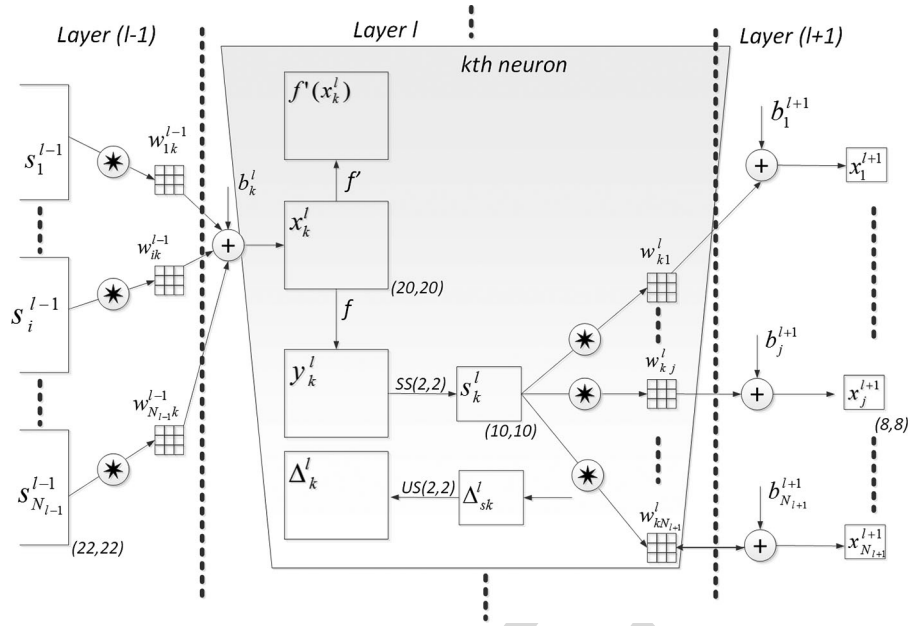

Fig. 2. Adaptive CNN implementation.

We aim that the number of hidden CNN layers can be set to any number. This ability is possible in this implementation because the subsampling factor of the output CNN layer (the hidden CNN layer just before the first MLP layer) is automatically set to the dimensions of its input map, e.g., in Fig. 2, if the layer $l+1$ would be the output CNN layer, then the subsampling factors for that layer will be $ssx = ssy = 8$ since the input map dimension is 8×8 in this sample illustration. Besides the subsampling, note that the dimension of the input maps will gradually decrease due to the convolution without zero padding, i.e., in Fig. 2, the dimension of the neuron output is 22×22 at the layer $l-1$ that is reduced to 20×20 at the layer l. As a result of this, the dimension of the input maps of the current layer is reduced by (Kx–1, Ky–1), where Kx and Ky are the width and height of the kernel, respectively.

A. Intra-BP Within a CNN Neuron: $\Delta_k^l \longleftarrow \Delta s_k^l$

The BP among MLP layers and from the first MLP layer to the output CNN layer is covered in Appendix. Once the first BP is performed from the next layer, $l+1$, to the current layer, l, then we can further BP it to the input delta. Let zero order upsampled map be: $us_k^l = up_{ssx,ssy}(s_k^l)$, then one can write

$$\begin{aligned}\Delta_k^l = \frac{\partial E}{\partial x_k^l} &= \frac{\partial E}{\partial y_k^l}\frac{\partial y_k^l}{\partial x_k^l} = \frac{\partial E}{\partial us_k^l}\frac{\partial us_k^l}{\partial y_k^l} f'(x_k^l) \\ &= up(\Delta s_k^l)\, \beta\, f'(x_k^l) \end{aligned} \quad (2)$$

where $\beta = (ssx.ssy)^{-1}$ since each pixel of s_k^l was obtained by averaging $ssx.ssy$ number of pixels of the intermediate output y_k^l. If maximum pooling is used instead of averaging, then (2) should be adapted accordingly.

B. Inter-BP Among CNN Layers: $\Delta s_k^l \xleftarrow{\Sigma} \Delta_l^{l+1}$

Recall the basic rule of BP: if the output of the kth neuron at layer l, contributes a neuron i in the next level with a weight, w_{ki}^l, that next layer neuron's delta, Δ_i^{l+1}, will contribute with the same weight to form Δ_k^l of the neuron in the previous layer l. This means

$$\frac{\partial E}{\partial s_k^l} = \Delta s_k^l \xleftarrow{\Sigma} \Delta_i^{l+1}\ \forall i \in \{1, N_{l+1}\} \quad (3)$$

Specifically

$$\frac{\partial E}{\partial s_k^l} = \Delta s_k^l = \sum_{i=1}^{N_{l+1}} \frac{\partial E}{\partial x_i^{l+1}}\frac{\partial x_i^{l+1}}{\partial s_k^l} = \sum_{i=1}^{N_{l+1}} \Delta_i^{l+1}\frac{\partial x_i^{l+1}}{\partial s_k^l} \quad (4)$$

where

$$x_i^{l+1} = \ldots + s_k^l * w_{ki}^l + \cdots \quad (5)$$

where "$*$" is the regular conv2D(.,.) operator without zero padding. It is obviously hard to compute the derivative directly from the 2-D convolution. Instead let us focus on the contribution of a single output pixel, $s_k^l(m,n)$, to the pixels of the next layer map, $x_i^{l+1}(m,n)$, assuming a 3×3 kernel

$$\begin{aligned} x_i^{l+1}(m-1,n-1) &= \ldots + s_k^l(m,n).w_{ki}^l(2,2) + \ldots \\ x_i^{l+1}(m-1,n) &= \ldots + s_k^l(m,n).w_{ki}^l(2,1) + \ldots \\ &\ldots\ldots \\ x_i^{l+1}(m+1,n+1) &= \ldots + s_k^l(m,n).w_{ki}^l(0,0) + \ldots \end{aligned} \quad (6)$$

This is illustrated in Fig. 3, where the role of an output pixel, $s_k^l(m,n)$, over two pixels of the next layer's input neuron's pixels $x_i^{l+1}(m-1,n-1)$and $x_i^{l+1}(m+1,n+1)$ can be seen.

Considering the pixel as a MLP neuron connected to other MLP neurons in the next layer, according to the basic rule of

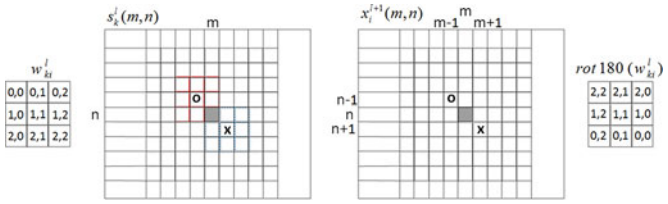

Fig. 3. Single pixel's contribution of the output $s_k^l(m,n)$ to the two pixels of the x_i^{l+1} using a 3×3 kernel.

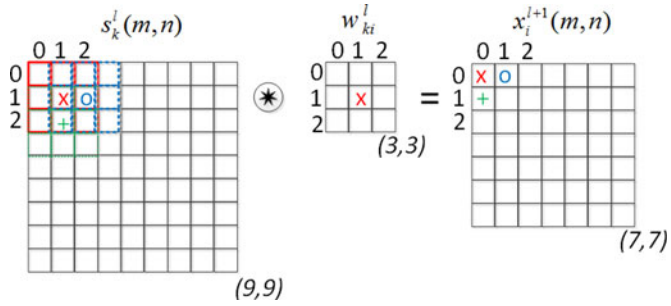

Fig. 4. Convolution of the output of the current layer neuron s_k^l and kernel w_{ki}^l to form the input of the ith neuron x_i^{l+1} at the next layer $l+1$.

BP one can then easily write the delta of $s_k^l(m,n)$ as

$$\frac{\partial E}{\partial s_k^l}(m,n) = \Delta s_k^l(m,n) = \sum_{i=1}^{N_{l+1}} \left(\sum_{r=-1}^{1} \sum_{t=-1}^{1} \Delta_i^{l+1}(m+r,n+t) \cdot w_{ki}^l(1-r,1-t) \right) \quad (7)$$

generalizing it for all pixels of the Δs_k^l yields

$$\Delta s_k^l = \sum_{i=1}^{N_{l+1}} \text{conv2D}z\left(\Delta_i^{l+1}, \text{rot180}(w_{ki}^l)\right) \quad (8)$$

where rot180(.) rotates the kernel, w_{ki}^l, 180°, and then conv2Dz(.,.) performs a *full* convolution with zero padding by (Kx–1, Ky–1) zeros to each boundary of the Δ_i^{l+1} in order to achieve equal dimensions (width and height) for Δs_k^l and Δ_i^{l+1} with the s_k^l.

C. Computation of the Weight (Kernel) and Bias Sensitivities

As in the regular BP on MLPs, the delta of the ith neuron at layer $l+1$, Δ_i^{l+1} will be used to update the bias of that neuron, and all weights of the neurons in the previous layer connected to that neuron as given in (19):

$$x_i^{l+1} = b_i^{l+1} + \cdots + y_k^l w_{ki}^l + \cdots$$
$$\therefore \frac{\partial E}{\partial w_{ki}^l} = y_k^l \, \Delta_i^{l+1} \text{ and } \frac{\partial E}{\partial b_i^{l+1}} = \Delta_i^{l+1}. \quad (9)$$

Recall the update rule: The sensitivity of the weight connecting the kth neuron in the current layer to the ith neuron in the next layer depends on the output of the current layer neuron, and the delta of the next layer neuron. For CNN layer neurons, we need to follow a similar approach to find out weight and bias sensitivities. Fig. 4 illustrates the convolution of the output of the current layer neuron s_k^l and kernel w_{ki}^l to form the input of the ith neuron x_i^{l+1} at the next layer $l+1$.

So now, we can focus on the contribution of each kernel element over the output. The following expressions can be written for the sample 2-D convolution shown in Fig. 4:

$$\begin{aligned}
x_i^{l+1}(0,0) &= \cdots + w_{ki}^l(0,0)s_k^l(0,0) + w_{ki}^l(0,1)s_k^l(0,1) \\
&\quad + w_{ki}^l(1,0)s_k^l(1,0) + \cdots \\
x_i^{l+1}(0,1) &= \cdots + w_{ki}^l(0,0)s_k^l(0,1) + w_{ki}^l(0,1)s_k^l(0,2) \\
&\quad + w_{ki}^l(1,0)s_k^l(1,1) + \cdots \\
x_i^{l+1}(1,0) &= \cdots + w_{ki}^l(0,0)s_k^l(1,0) + w_{ki}^l(0,1)s_k^l(1,1) \\
&\quad + w_{ki}^l(1,0)s_k^l(2,0) + \cdots \\
x_i^{l+1}(m,n) &= \cdots + w_{ki}^l(0,0)s_k^l(m,n) + w_{ki}^l(0,1) \\
&\quad \times s_k^l(m,n+1) + w_{ki}^l(1,0)s_k^l(m+1,n) + \cdots \\
x_i^{l+1}(m,n) &= \sum_{r=-1}^{1} \sum_{t=-1}^{1} w_{ki}^l(r+1,t+1) \\
&\quad \times s_k^l(m+r+1,n+t+1). \quad (10)
\end{aligned}$$

Since each weight (kernel) element is used (shared) in common to form each neuron input, $x_i^{l+1}(m,n)$, the derivative will be the cumulation of delta-output product for all pixels, i.e.,

$$\frac{\partial E}{\partial w_{ki}^l(r,t)} = \sum_m \sum_n \Delta_i^{l+1}(m,n) s_k^l(m+r,n+t)$$
$$\Rightarrow \frac{\partial E}{\partial w_{ki}^l} = \text{conv2D}(s_k^l, \Delta_i^{l+1}). \quad (11)$$

Similarly, the bias of this neuron, b_k^l, contributes to all pixels in the image (same bias shared among all pixels), so its sensitivity will be the cumulation of individual pixel sensitivities i.e.,

$$\frac{\partial E}{\partial b_k^l} = \sum_m \sum_n \frac{\partial E}{\partial x_k^l(m,n)} \frac{\partial x_k^l(m,n)}{\partial b_k^l} = \sum_m \sum_n \Delta_k^l(m,n). \quad (12)$$

As a result, the iterative flow of the BP can be stated as follows.

1) Initialize weights [usually randomly, U(–a, a)].
2) For each BP iteration DO:
   a) For each item (or a group of items or all items) in the dataset, DO:

i) *FP:* FP from the input layer to the output layer to find outputs of each neuron at each layer $y_i^l \; \forall i \in [1, N_l]$ and $\forall l \in [1, L]$.
ii) *BP:* Compute delta error at the output layer and BP it to first hidden layer to compute the delta errors $\Delta_k^l \; \forall k \in [1, N_l]$ and $\forall l \in [2, L-1]$.
iii) *PP:* Postprocess to compute the weight and bias sensitivities using (11), (12), and (25).
iv) *Update:* Update the weights and biases with the (cumulation of) sensitivities found in (c) scaled with the learning factor ε:

$$w_{ik}^{l-1}(t+1) = w_{ik}^{l-1}(t) - \varepsilon \frac{\partial E}{\partial w_{ik}^{l-1}}$$
$$b_k^l(t+1) = b_k^l(t) - \varepsilon \frac{\partial E}{\partial b_k^l} \qquad (13)$$

B. Changes for the Adaptive 1-D CNN Implementation

There are only some minor differences between the 2-D and 1-D CNNs. The main difference is obviously the 1-D arrays used in each neuron for its kernels (weights), input and output elements for both FP and BP rather than 2-D matrices. Therefore, during both FP and BP runs, 1-D array manipulations such as conv1D and reverse will be performed instead of 2-D matrix operations, such as conv2D and rot180. The 2-D parameters for kernel size (*Kx, Ky*) and subsampling (*ssx, ssy*) are now single scalars *K* and *ss*, respectively. If present, the MLP layers of the 1-D CNN are identical to the 2-D counterpart, and, therefore, same equations can be used for both FP and BP on those layers. The expression of the 2-D FP given in (26) can now be written as

$$x_k^l = b_k^l + \sum_{i=1}^{N_{l-1}} \text{conv1}D(w_{ik}^{l-1}, s_i^{l-1}). \qquad (14)$$

The inter-BP delta error of the output s_k^l given in (8) can now be expressed for 1-D as

$$\Delta s_k^l = \sum_{i=1}^{N_{l+1}} \text{conv1}Dz\left(\Delta_i^{l+1}, \text{rev}(w_{ki}^l)\right) \qquad (15)$$

where rev(.) reverses the array, and conv1*Dz*(.,.) performs full convolution in 1-D with *K*–1 zero padding. The intra-BP is also identical to the one for 2-D CNNs, as expressed in (2), while $\beta = ss^{-1}$ and a 1-D upsampling operator is now used. Finally, the weight and bias sensitivities given in (11) and (12) for 2-D can now be expressed for 1-D CNNs as

$$\frac{\partial E}{\partial w_{ki}^l} = \text{conv1D}(s_k^l, \Delta_i^{l+1})$$
$$\frac{\partial E}{\partial b_k^l} = \sum_n \Delta_k^l(n). \qquad (16)$$

IV. Experimental Results

In this section, we shall first present the experimental setup for the test and evaluation of the proposed patient-specific ECG classification approach. We shall then present the overall results obtained from the ECG classification experiments and perform comparative evaluations against several state-of-the-art techniques in this field. The robustness of the proposed system against variations of signal resolution (hence, the CNN configuration) and raw data representation will then be evaluated. Finally, the computational complexity of the proposed method for both training and classification will be evaluated in detail.

A. Experimental Setup

As mentioned earlier, in order to evaluate the effects of different resolutions and raw data representations particularly in the transform domain, we used two alternatives for each case. The beats are represented in both 64 and 128 samples centered on the R-peak, and we used the FFT representation (magnitude and phase) of each beat as in the extended data representation. We purposefully used a simple 1-D CNN in all experiments with only three CNN layers and two MLP layers, in order to achieve the utmost computational efficiency for both training and particularly for real-time classification. On top of this, we aim to demonstrate that deep learners are not indeed needed to achieve a superior ECG classification performance. The 1-D CNN used in all experiments has 32 and 16 neurons on the first- and second-hidden CNN layers and ten neurons on the hidden MLP layer. The output (MLP) layer size is 5 which is the number of beat classes and the input (CNN) layer size is either 2 (base*)* or 4 (extended*)* according to the choice of raw data representation. For 64 and 128 sample beat representations, the kernel sizes are set to 9 and 15, and the subsampling factors are set to 4 and 6, respectively. As a result, in the proposed adaptive CNN implementation, the subsampling factors for the last CNN layers are automatically set to 6 and 5, respectively.

For all experiments, we employ a shallow training: the maximum number of BP iterations is set to 50, and another stopping criterion is the minimum train classification error level that is set to 3% to prevent overfitting. Therefore, the training will terminate if either of the criteria is met. We initially set the learning factor ε as 0.001, and applied a global adaptation during each BP iteration; if the train mean-squared error (MSE) decreases in the current iteration, we slightly increase ε by 5%; otherwise, we reduce it by 30% for the next iteration. As BP is a deterministic gradient-descent optimization technique, which makes it quite dependent to the initial (random) setting of the network parameters (kernels, weights, and biases), we performed ten individual BP runs for each patient in the database, and the average classification performance is reported.

B. Base Classification Performance Evaluation

We performed classification experiments on 44 records of the MIT/BIH arrhythmia database, containing a total of 100389 beats to be classified into five heartbeat types according to the AAMI recommendation [13]. For training the 1-D CNNs, both common and patient-specific training patterns are used; the common part of the training dataset contains a total of 245 representative beats, including 75 from each type-N, type-S, and type-V beats, and all (13) type-F and (7) type-Q beats, randomly

TABLE I
CONFUSION MATRICES OF THE ECG BEAT CLASSIFICATION RESULTS FOR ALL 44 RECORDS (BOTTOM) AND FOR THE 24 TEST RECORDS (TOP) IN THE MIT/BIH ARRHYTHMIA DATABASE

		Classification Result				
Ground Truth		N	S	V	F	Q
	N	40963F (40532)	807 (776)	350 (382)	67 (56)	4 (20)
	S	625 (672)	1440 (1441)	149 (197)	14 (5)	1 (5)
	V	114 (392)	69 (299)	4247 (4022)	39 (75)	2 (32)
	F	82 (164)	4 (26)	70 (46)	497 (378)	0 (2)
	Q	6 (6)	2 (0)	5 (1)	0 (1)	0 (0)

		Classification Result				
Ground Truth		N	S	V	F	Q
	N	73539 (73019)	824 (991)	368 (513)	69 (98)	5 (29)
	S	837 (686)	1568 (1568)	178 (205)	15 (5)	2 (6)
	V	230 (462)	72 (333)	5277 (4993)	39 (79)	4 (32)
	F	92 (168)	4 (28)	73 (48)	503 (379)	0 (2)
	Q	31 (8)	2 (1)	5 (3)	0 (1)	4 (1)

The previous results from [16] are shown in parentheses.

TABLE II
VEB AND SVEB CLASSIFICATION PERFORMANCE OF THE PROPOSED METHOD AND COMPARISON WITH FOUR MAJOR ALGORITHMS FROM THE LITERATURE

Methods	VEB				SVEB			
	Acc	*Sen*	*Spe*	*Ppr*	*Acc*	*Sen*	*Spe*	*Ppr*
Hu *et al.* [10][1]	94.8	78.9	96.8	75.8	N/A	N/A	N/A	N/A
Jiang and Kong [15] [1]	98.8	94.3	99.4	95.8	**97.5**	74.9	98.8	78.8
Ince *et al.* [16][1]	97.9	90.3	98.8	92.2	96.1	**81.8**	98.5	63.4
Proposed[1]	**98.9**	**95.9**	**99.4**	**96.2**	96.4	68.8	**99.5**	**79.2**
Jiang and Kong [15] [2]	98.1	86.6	**99.3**	**93.3**	**96.6**	50.6	**98.8**	**67.9**
Ince *et al.* [16][2]	97.6	83.4	98.1	87.4	96.1	62.1	98.5	56.7
Proposed[2]	**98.6**	**95**	98.1	89.5	96.4	**64.6**	98.6	62.1
Ince *et al.* [16] [3]	98.3	84.6	98.7	87.4	97.4	**63.5**	99.0	53.7
Proposed[3]	**99**	**93.9**	**98.9**	**90.6**	**97.6**	60.3	**99.2**	**63.5**

Best results are highlighted.
[1]The comparison results are based on 11 common recordings for VEB detection and 14 common recordings for SVEB detection.
[2]The VEB and SVEB detection results are compared for 24 common testing records only.
[3]The VEB and SVEB detection results of the proposed system for all training and testing records.

sampled from each class from the first 20 records (picked from the range 100 to 124) of the MIT/BIH database, and the patient-specific training data include the beats from the first 5 min of the corresponding patient's ECG record. Patient-specific 1-D CNN networks are trained with a total of 245 common training beats, and a variable number of patient-specific beats depending on the patient's heart rate, so only less than 1% of the total beats are used for training. The remaining beats (25 min) of each record, in which 24 out of 44 records are completely new to the classifier, are used as test patterns for performance evaluation.

Classification performance is measured using the four standard metrics found in the literature [10]: classification accuracy (*Acc*), sensitivity (*Sen*), specificity (*Spe*), and positive predictivity (*Ppr*). While accuracy measures the overall system performance over all classes of beats, the other metrics are specific to each class, and they measure the ability of the classification algorithm to distinguish certain events (i.e., VEBs or SVEBs) from nonevents (i.e., non-VEBs or non-SVEBs). The respective definitions of these four common metrics using true positive (*TP*), true negative (*TN*), false positive (*FP*), and false negative (*FN*) are as follows: *Accuracy* is the ratio of the number of correctly classified patterns to the total number of patterns classified, $Acc = (TP+TN)/(TP+TN+FP+FN)$; *Sensitivity* is the rate of correctly classified events among all events, $Sen = TP/(TP+FN)$; *Specificity* is the rate of correctly classified nonevents among all nonevents, $Spe = TN/(TN+FP)$; and *Positive Predictivity* is the rate of correctly classified events in all detected events, $Ppr = TP/(TP+FP)$. Since there is a large variation in the number of beats from different classes in the training/testing data (i.e., 39465/50354 type-N, 1277/5716 type-V, and 190/2571 type-S beats), sensitivity, specificity, and positive predictivity are more relevant performance criteria for medical diagnosis applications.

TABLE III
CONFUSION MATRIX FOR THE ECG BEAT CLASSIFICATION OF THE PATIENT 202

		Classification Result				
Ground Truth		N	S	V	F	Q
	N	1435	258	10	0	0
	S	11	15	28	0	0
	V	4	1	10	0	0
	F	1	0	0	0	0
	Q	0	0	0	0	0

For the base performance evaluation, we shall consider the *base* raw data represented by 128 samples (both the single beat and the beat trio). The results of the other three alternatives from the extended raw data and 64 samples representation over the other 1-D CNN setup will be considered to demonstrate the robustness of the system against the variations on raw data representations, signal resolution, and CNN configurations. For the proposed approach and the previous state-of-the-art method in [16], Table I presents the confusion matrices of ECG beat classification results for all 44 records and the 24 records of the test dataset. To perform a more extensive and accurate comparative performance evaluation, the base performance of the proposed system is compared with the three existing algorithms, [10], [15], and [16], all of which comply with the AAMI standards. Although P. de Chazal in [12] and [14] also comply with the AAMI standards, since they used different training and test data, it is not possible to directly compare their results with the proposed method. In accordance with the AAMI recommendations, the problem of VEB and SVEB detection is considered individually and results are summarized in Table II. The benchmark MIT/BIH database is partitioned into three evaluation datasets. For VEB detection, the dataset 1 contains 11 test recordings (200, 202, 210, 213, 214, 219, 221, 228, 231, 233, and 234), and for SVEB detection, comparison results are based on 14 common recordings (with the addition of records 212, 222, and 232). Dataset 1 is common for all competing methods. Dataset 2 is the test partition of the benchmark database containing 24 records (200 and onward). Three methods (the proposed, [15], and [16]) are tested on this dataset. Finally, the dataset 3 is the

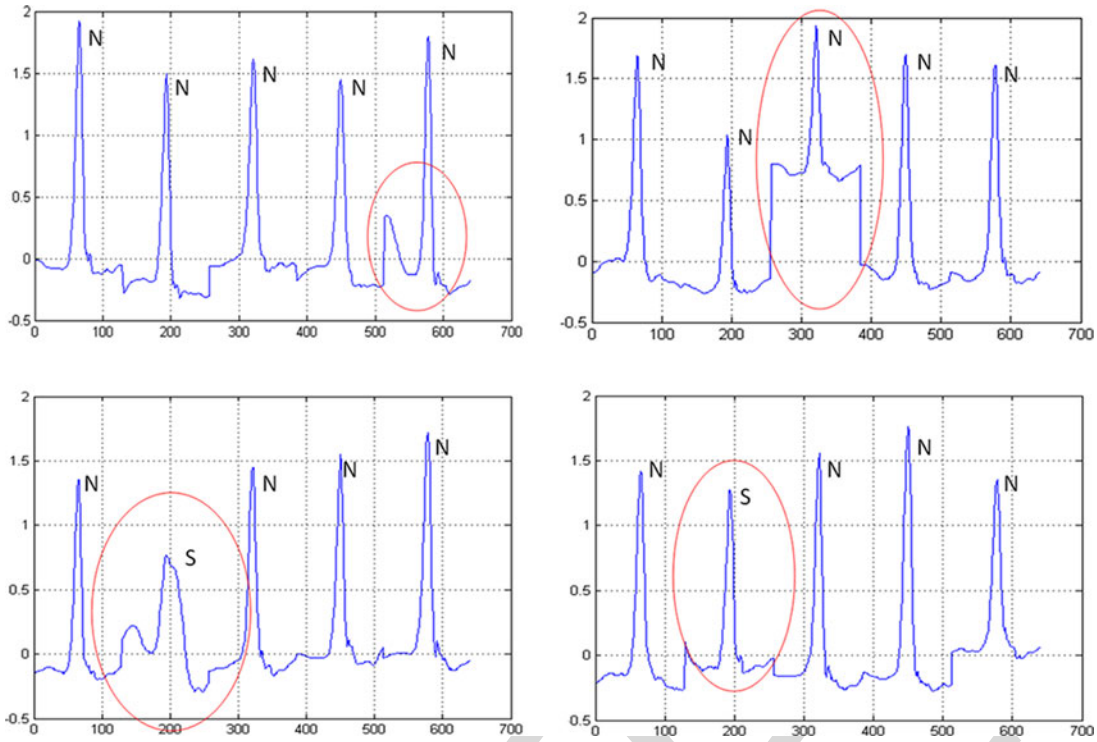

Fig. 5. Four 5-beats interval from the test section of the patient 202's ECG record with the ground truth labels.

entire database with all records over which two methods, the proposed and [16] are tested.

Several interesting observations can be made from the results in Table II. First, for SVEB detection, sensitivity and positive predictivity rates are comparably lower than VEB detection, while a high-specificity performance is achieved. The reason for the worse classifier performance in detecting SVEBs is that SVEB class is underrepresented in the training data, and, hence, more SVEB beats are misclassified as normal beats. Moreover, on several patients, particularly on the test dataset (dataset 2) both the patient specific data from the first 5-min interval and the common data of 200 beats extracted randomly from the training dataset do not successfully characterize most of the S beats and some of the V beats (e.g., patients 201, 202, 209, 222, and 232). Take, for instance, the confusion matrix given for patient 202 in Table III, where 258 normal beats are misclassified as S beats and 28 S beats are misclassified as V beats. The four 5-beat intervals from the test section of this patient's ECG record are shown in Fig. 5. Such anomalies shown in the plots on N and S beats do not in fact exist in the training dataset of this patient, and, hence, the classifier can easily misclassify them. Particularly, the plot in the bottom-left was misclassified as a V beat due to its morphological anomaly that was learned as a V beat, and the S beat in the bottom-right was misclassified as N beat due to its strong resemblance to the pattern of N beats. This is why, the selection of the common data is of utmost importance and rather than random, the selection should be performed with care to cover as much representative beats as possible to minimize such an inevitable source of classification errors.

In the next section, we shall test the robustness of the proposed approach against the variations of the signal resolution along with the network configuration and raw data representation to validate whether the performance can significantly deteriorate.

C. Robustness

Over each dataset partitioning, we evaluate the proposed approach by varying the beat resolution (64 versus 128 samples) along with the CNN parameters and also the raw data representation (base versus extended). We observed that mostly comparable performances with the base performance level are achieved for all measures on both VEB and SVEB classification, while the performance drop becomes somewhat significant (∼8%) only for the positive predictivity of the SVEB classification. Other measures are usually slightly comparable with the base performance. It is quite evident that the effects of such variations over the classification accuracy are usually insignificant, especially for VEB classification. Therefore, other raw data, resolutions, and CNN parameter variants can be conveniently used within the proposed approach without sacrificing from the base performance level. The only cost using the base (128 samples per beat) setting rather than the 64 samples resolution is the increased computational complexity due to the larger 1-D convolutions during both BP and FP. This will be investigated in detail in next section.

D. Computational Complexity

We implemented the proposed adaptive 1-D CNN using C++ over MS Visual Studio 2013 in 64 bit. This is a non-GPU implementation; however, Intel OpenMP API is used to obtain multiprocessing with a shared memory. The experiments are performed on a computer with I7-4700MQ at 2.4 GHz

(eight CPUs) and 16-Gb memory. In theory, this should yield to 8× speed improvement, but, in practice, the observed speed improvement was between 4.8× and 5×.

Considering the baseline implementation (with base raw data with 128 samples beat resolution), the average time for 1 BP iteration per beat is about 4.24 m·sec with this setting. In a single CPU implementation, it becomes 21.2 m · sec. Considering a complete BP run with maximum 50 iterations over a patient's ECG training dataset with maximum 600 beats (maximum 400 patient specific + 200 common), this means that the maximum time for training would be $600 \times 50 \times 21.2$ m · sec = 10.6 min. The average training time for [16] was about 24 min, including the feature extraction and postprocessing operations. In our implementation with OpenMP, this was less than 2 min per patient, which is an insignificant time for training the system which will be performed only once per patient. For the 64 samples beat resolution, the average time for 1 BP iteration per beat drops down to 3.93 m · sec.

The most important advantage of the proposed system is its significantly low computational cost for the beat classification. Specifically, for the single-CPU implementation, the total time for a FP of a single beat to obtain the class vector is about 0.58 and 0.74 m · sec for 64 and 128 samples beat resolutions, respectively. Although there is now a significant decrease in the computational cost for using 64 samples beat resolution, note that this speed is still more than 1000× faster than the real-time requirement.

The main limitation of the proposed approach is that the characterization of the crucial anomaly beats such as the S beats. As discussed earlier, neither the patient specific data nor the common data that are randomly collected guarantees to represent such anomaly beats properly, and if they are not included in the training dataset, the system may entirely or partially fail to classify them. Furthermore, the proposed approach like the prior works in this domain do not support incremental or active learning, which hinders the ability to adapt to the changes of the patient's heartbeat patterns.

V. Conclusion

In this study, we proposed a patient-specific ECG heartbeat classifier with an adaptive implementation of 1-D CNNs that are able to fuse the two major blocks of the traditional ECG classification into a single learning body: feature extraction and classification. Such a compact implementation, for each patient over a simple CNN, not only negates the necessity to extract handcrafted manual features, or any kind of pre- and postprocessing, also makes it a primary choice for a real-time implementation for heart monitoring and anomaly detection. Besides the speed and computational efficiency achieved, the proposed method only requires 1-D convolutions (multiplications and additions) that make any hardware implementation simpler and cheaper. In addition to that, once a dedicated CNN is trained for an individual patient, it can solely be used to classify his/her long ECG records such as Holter registers in a fast and accurate manner.

The results of the classification experiments, which are performed over the benchmark MIT/BIH arrhythmia database show that for both test datasets (1 and 2), the base performance level of the proposed approach in both VEB and SVEB detection is comparable or better than the competing methods for most of the measures. Over the entire dataset (dataset 3), it has the highest performance measures except the SVEB sensitivity. We also observed that variations on the beat resolution with a different CNN settings and raw data representations are not degrading and can be desirable for even lower computational complexity. As a result, the proposed approach achieves the main design objectives, i.e., maintaining a fast, robust, and patient-specific system with a superior classification performance. As a future work, we are planning to design the hardware implementation of the proposed approach.

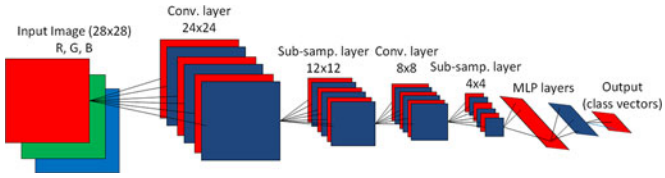

Fig. 6. Overview of a sample conventional CNN (top).

Appendix

A. CNN Overview

CNNs are simple feedforward ANNs that are simple models of mammalian visual cortex and the same themes have been revealed in the past ten years by the auditory neuroscience that these same design paradigms can be found in the primary and belt auditory areas of the cortex in a number of different animals. Fig. 6 shows a typical CNN structure which is designed for 28 × 28 pixel images. There are subsampling layers between each convolutional layer, which decimate the so-called feature maps of the previous layers neurons. In this sample illustration, the kernel size is set to 5, and the subsampling factors for both dimensions are set to 2. The input layer has the input image (R, G, B) color channels as the feature maps, and after sufficient number of subsampling, the last subsampling layer reaches a scalar (1-D) neurons. Following, CNNs contain fully connected layers that have the same structure as the MLPs.

In order to accomplish this setup, there are several parameters to be set for such typical CNNs: width and height of the input image, kernel (filter) dimensions at each level (usually fixed for each level), the topology of the CNN (number of CNN and MLP hidden layers and number of neurons at each layer), and subsampling factors at each level (usually fixed for each level). Moreover, the input image dimensions should be set according to the number of CNN layers, kernel size, and subsampling factors so that the output CNN layer can reach to a scalar 1 × 1 feature map. One can only train the CNN, once all these parameters are properly fixed in advance. To address these drawbacks, we shall present an adaptive CNN implementation next.

B. Conventional BP

1) BP at MLP layers: In our adaptive CNN implementation, the MLP layers are optional, and if present, the BP operation

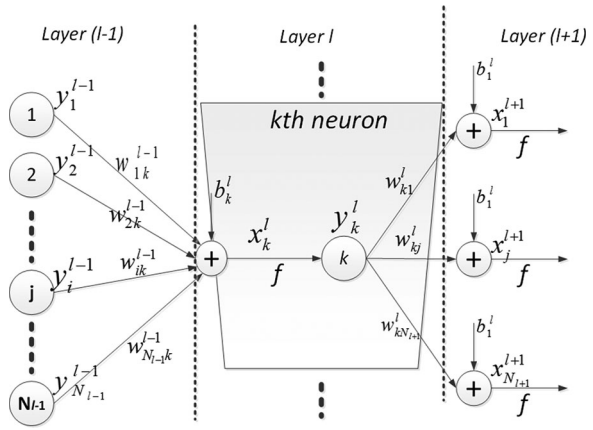

Fig. 7. Three consecutive MLP layers.

starts from the output MLP layer toward the inner layers. Consider a generic MLP illustration in Fig. 7 showing the connections between three layers, $l-1$ to $l+1$ to and from the kth neuron at layer l. The FP of the outputs at layer l–1 toward the output of the kth neuron at layer l can be expressed as follows:

$$x_k^l = b_k^l + \sum_{i=1}^{N_{l-1}} w_{ik}^{l-1} y_i^{l-1} \text{ and } y_k^l = f(x_k^l). \tag{17}$$

Let $l = 1$ and $l = L$ be the input and output layers, respectively. For an input vector p, and its corresponding output vector, $[y_1^L, \ldots, y_{N_L}^L]$, let $[t_1, \ldots, t_{N_L}]$ be the target class vector. The MSE in the output layer can then be expressed as

$$E = E(y_1^L, \ldots, y_{N_L}^L) = \sum_{i=1}^{N_L} \left(y_i^L - t_i\right)^2. \tag{18}$$

We are interested to find out the derivative of this error with respect to an individual weight (connected to that neuron, k) w_{ik}^{l-1}, and bias of the neuron k, b_k^l, so that we can perform gradient descent method to minimize the error accordingly

$$\begin{aligned} \frac{\partial E}{\partial w_{ik}^{l-1}} &= \frac{\partial E}{\partial x_k^l}\frac{\partial x_k^l}{\partial w_{ik}^{l-1}} = \frac{\partial E}{\partial x_k^l} y_i^{l-1} \\ \frac{\partial E}{\partial b_k^l} &= \frac{\partial E}{\partial x_k^l}\frac{\partial x_k^l}{\partial b_k^l} = \frac{\partial E}{\partial x_k^l}. \end{aligned} \tag{19}$$

Both derivatives depend on the sensitivities of the error to the input x_k^l. These sensitivities are usually called as *delta* errors. Let $\Delta_k^l = \frac{\partial E}{\partial x_k^l}$ be the delta error of the kth neuron at layer l. Now, we can write the *delta* error by one step BP from the output of that neuron y_k^l:

$$\Delta_k^l = \frac{\partial E}{\partial x_k^l} = \frac{\partial E}{\partial y_k^l}\frac{\partial y_k^l}{\partial x_k^l} = \frac{\partial E}{\partial y_k^l} f'(x_k^l). \tag{20}$$

This means that the moment we found the sensitivities of the error to the output $\frac{\partial E}{\partial y_k^l}$, we can then find the *delta* error. For the output layer $l = L$, we know both terms

$$\Delta_k^L = \frac{\partial E}{\partial x_k^L} = f'(x_k^L)\left(y_k^L - t_k\right). \tag{21}$$

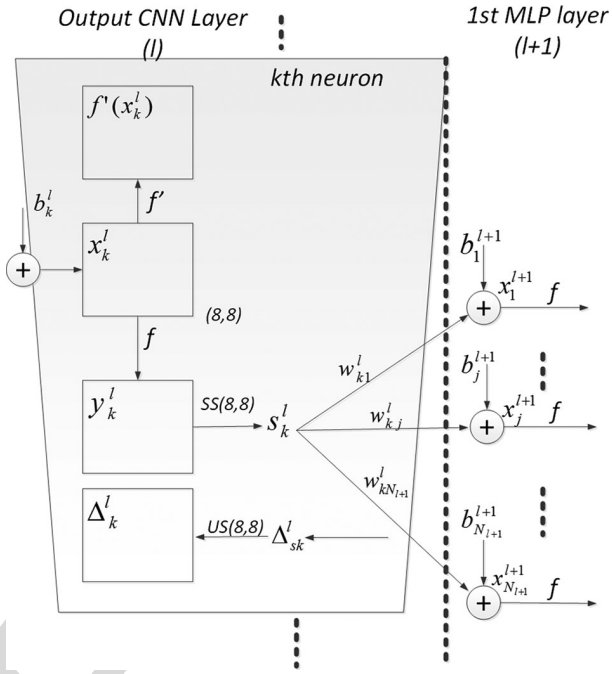

Fig. 8. Output CNN layer connected to the first MLP layer.

Now Fig. 7 considers the output of the previous layer neuron's output y_k^l, which contributes all the neurons' input in the next layer, i.e.,

$$\begin{aligned} x_1^{l+1} &= \ldots + w_{k1}^l y_k^l + \ldots \\ &\ldots \\ x_i^{l+1} &= \ldots + w_{ki}^l y_k^l + \ldots \\ &\ldots \\ x_{N_{l+1}}^{l+1} &= \ldots + w_{kN_{l+1}}^l y_k^l + \ldots . \end{aligned} \tag{22}$$

Therefore, this basically means that the output of the kth neuron in the previous layer y_k^l contributes to the input of the neurons of the current layer with individual weights w_{ki}^l. With this in mind, one can write the sensitivities of the error to the output $\frac{\partial E}{\partial y_k^l}$ as follows:

$$\frac{\partial E}{\partial y_k^l} = \sum_{i=1}^{N_{l+1}} \frac{\partial E}{\partial x_i^{l+1}}\frac{\partial x_i^{l+1}}{\partial y_k^l} = \sum_{i=1}^{N_{l+1}} \Delta_i^{l+1} w_{ki}^l. \tag{23}$$

Interestingly, the same weights are now used to BP the delta errors to the output sensitivity of that neuron, and we already know how to get the delta of that neuron Δ_k^l, from this sensitivity (in (20)), which leads to the generic equation of the BP of the deltas

$$\Delta_k^l = \frac{\partial E}{\partial x_k^l} = \frac{\partial E}{\partial y_k^l} f'(x_k^l) = f'(x_k^l) \sum_{i=1}^{N_{l+1}} \Delta_i^{l+1} w_{ki}^l. \tag{24}$$

This means that the delta of the kth neuron at layer l, Δ_k^l, will be formed by all deltas of the next layer, Δ_i^{l+1}, weighted by the connections from k to i, w_{ki}^l. In other words, if the output of the kth neuron at layer l contributes to a neuron i in the next

level with a weight w_{ki}^l, then next layer neuron's delta Δ_i^{l+1} will contribute with the same weight to form Δ_k^l of the neuron in the previous layer l.

Once all the deltas in each layer are formed by BP, then weights and bias of each neuron can be updated by the gradient descent method. Specifically, the delta of the kth neuron at layer l, Δ_k^l will be used to update the bias of that neuron and all weights of the neurons in the previous layer connected to that neuron as given in (19)

$$\frac{\partial E}{\partial w_{ik}^{l-1}} = \Delta_k^l y_i^{l-1} \quad \text{and} \quad \frac{\partial E}{\partial b_k^l} = \Delta_k^l. \tag{25}$$

In simple words, the sensitivity of the weight connecting the ith neuron in the previous layer to the kth neuron in the current layer depends on the output of the previous layer neuron and the delta of the current layer neuron.

2) BP From the MLP layer to the Output CNN Layer: As illustrated in Fig. 8, the output layer of CNN is connected to the first MLP layer, and, hence, the outputs of this layer CNN neurons are scalars. In other words, s_k^l and of course, Δs_k^l are now all scalars and to achieve this recall that the subsampling factors *ssx* and *ssy*, in this particular layer all are set to the dimensions of the input map. Similarly, the weights of this CNN layer neurons w_{ki}^l are also all scalar and instead of convolution, scalar multiplication is performed as in a regular MLP. So from MLP layer to the CNN layer, the regular BP is simply performed as in (23):

$$\frac{\partial E}{\partial s_k^l} = \Delta s_k^l = \sum_{i=1}^{N_{l+1}} \frac{\partial E}{\partial x_i^{l+1}} \frac{\partial x_i^{l+1}}{\partial s_k^l} = \sum_{i=1}^{N_{l+1}} \Delta_i^{l+1} w_{ki}^l. \tag{26}$$

REFERENCES

[1] R. Hoekema *et al.*, "Geometrical aspects of the interindividual variability of multilead ECG recordings," *IEEE Trans. Biomed. Eng.*, vol. 48, no. 5, pp. 551–559, May 2001.

[2] K. Minami *et al.*, "Real-time discrimination of ventricular tachyarrhythmia with Fourier-transform neural network," *IEEE Trans. Biomed. Eng.*, vol. 46, no. 2, pp. 179–185, Feb. 1999.

[3] Inan *et al.*, "Robust neural-network based classification of PVCs using wavelet transform and timing interval features," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 12, pp. 2507–2515, Dec. 2006.

[4] X. Alfonso and T. Q. Nguyen, "ECG beat detection using filter banks," *IEEE Trans. Biomed. Eng.*, vol. 46, no. 2, pp. 192–202, Feb. 1999.

[5] J. L. Willems and E. Lesaffre, "Comparison of multigroup logisitic and linear discriminant ECG and VCG classification," *J. Electrocardiol.*, vol. 20, pp. 83–92, 1987.

[6] J. L. Talmon, *Pattern Recognition of the ECG*. Berlin, Germany: Akademisch Proefscrift, 1983.

[7] D. A. Coast *et al.*, "An approach to cardiac arrhythmia analysis using hidden Markov models," *IEEE Trans. Biomed. Eng.*, vol. 37, no. 9, pp. 826–836, Sep. 1990.

[8] S. Osowski *et al.*, "Support vector machine based expert system for reliable heartbeat recognition," *IEEE Trans.Biomed. Eng.*, vol. 51, no. 4, pp. 582–589, Apr. 2004.

[9] Y. H. Hu *et al.*, "Applications of artificial neural networks for ECG signal detection and classification," *J. Electrocardiol.*, vol. 26, pp. 66–73, 1994.

[10] Y. Hu *et al.*, "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.

[11] S. C. Lee, "Using a translation-invariant neural network to diagnose heart arrhythmia," in *Proc. IEEE Conf. Neural Inf. Process. Syst.*, Nov. 1989, pp. 2025–2026.

[12] P. de Chazal and R. B. Reilly, "A patient-adapting heartbeat classifier using ECG morphology and heartbeat interval features," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 12, pp. 2535–2543, Dec. 2006.

[13] *Recommended Practice for Testing and Reporting Performance Results of Ventricular Arrhythmia Detection Algorithms*. Arlington, VA, USA: Association for the. Advancement of Medical Instrumentation, 1987.

[14] P. de Chazal *et al.*, "Automatic classification of heartbeats using ECG morphology and heartbeat interval features," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 7, pp. 1196–1206, Jul. 2004.

[15] W. Jiang and S. G. Kong, "Block-based neural networks for personalized ECG signal classification," *IEEE Trans. Neural Networks*, vol. 18, no. 6, pp. 1750–1761, Nov. 2007.

[16] T. Ince *et al.*, "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.

[17] S. Kiranyaz *et al.*, *Multi-Dimensional Particle Swarm Optimization for Machine Learning and Pattern Recognition*. New York, NY, USA: Springer, Aug. 2013.

[18] M. Llamedo and J. P. Martinez, "An automatic patient-adapted ECG heartbeat classifier allowing expert assistance," *IEEE Trans. Biomed. Eng.*, vol. 59, no. 8, pp. 2312–2320, Aug. 2012.

[19] C. Li *et al.*, "Detection of ECG characteristic points using wavelet transforms," *IEEE Trans. Biomed. Eng.*, vol. 42, no. 1, pp. 21–28, Jan. 1995.

[20] T. Mar *et al.*, "Optimization of ECG classification by means of feature selection," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 8, pp. 2168–2177, Aug. 2011.

[21] D. H. Wiesel and T. N. Hubel, "Receptive fields of single neurones in the cat's striate cortex," *J. Physiol.*, vol. 148, pp. 574–591, 1959.

[22] R. Mark and G. Moody. MIT-BIH Arrhythmia Database Directory. [Online]. Available: http://ecg.mit.edu/dbinfo.html

[23] J. Pan and W. J. Tompkins, "A real-time QRS detection algorithm," *IEEE Trans. Biomed. Eng.*, vol. 32, no. 3, pp. 230–236, Mar. 1985.

[24] D. C. Ciresan *et al.*, "Deep big simple neural nets for handwritten digit recognition," *Neural Comput.*, vol. 22, no. 12, pp. 3207–3220, 2010.

[25] D. Scherer *et al.*, "Evaluation of pooling operations in convolutional architectures for object recognition," in *Proc. Int. Conf. Artif. Neural Netw.*, 2010, pp. 92–101.

[26] A. Krizhevsky *et al.*, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. Conf.*, 2012, pp. 1106–1114.

[27] 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.

Serkan Kiranyaz was born in Turkey in 1972. He received the B.S. degree from the Electrical and Electronics Department and the M.S. degree in signal and video processing from Bilkent University, Ankara, Turkey, in 1994, and 1996, respectively, and the Ph.D. degree from the Institute of Signal Processing, Tampere University of Technology, Tampere, Finland, in 2005.

He received the Docent title from the Institute of Signal Processing, Tampere University of Technology, in 2007. He was a Senior Researcher with Nokia Research Center, and later with Nokia Mobile Phones, Tampere. He was a Professor with the Signal Processing Department, Tampere University of Technology, during 2009 to 2015, and he held the Research Director position for the Department and also for the Center for Visual Decision Informatics, Finland. He is currently a Professor at the Department of Electrical Engineering, Qatar University, Doha, Qatar.

He has published two books, more than 35 journal papers on many IEEE Transactions and other high impact journals and around 80 papers in international conferences. His principal research interests include "intelligent systems" that comprises systems, devices, algorithms, software and processes requiring intelligent processing of, possibly "big" data, or sensor signals together with artificial reasoning. He made significant contributions on biosignal analysis, particularly EEG and ECG analysis and processing, classification and segmentation, signal processing and management, computer vision, scene analysis, automatic object segmentation, and machine learning strategies with applications to recognition, classification, multimedia retrieval, evolving systems and evolutionary machine learning, swarm intelligence, stochastic optimization, and deep learning.

Turker Ince received the B.S. degree from the Bilkent University, Ankara, Turkey, in 1994, the M.S. degree from the Middle East Technical University, Ankara, in 1996, and the Ph.D. degree from the University of Massachusetts Amherst (UMass Amherst), Amherst, MA, USA, in 2001, all in electrical engineering.

From 1996 to 2001, he was a Research Assistant with the Microwave Remote Sensing Laboratory, UMass Amherst. He was a Design Engineer with Aware, Inc., Boston, USA, from 2001 to 2004, and with Texas Instruments, Inc., Dallas, USA, from 2004 to 2006. In 2006, he joined the Faculty of Engineering and Computer Science, Izmir University of Economics, Izmir, Turkey, where he is currently an Associate Professor. His research interests include electromagnetic remote sensing and target recognition, radar signal processing, biomedical signal processing, artificial neural networks, and evolutionary optimization techniques.

Moncef Gabbouj (F'xx) received the B.S. degree in electrical engineering from Oklahoma State University, Stillwater, OK, USA, in 1985, and the M.S. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1986 and 1989, respectively. Q2

He held several visiting professorships with different universities, including The Hong Kong University of Science and Technology, Hong Kong; Purdue University; the University of Southern California; and the American University of Sharjah, UAE. He was the Head of the Department during 2002–2007, and was the Senior Research Fellow with the Academy of Finland in 1997–1998 and 2007–2008. He is currently the Academy of Finland Professor of signal processing at the Department of Signal Processing, Tampere University of Technology, Tampere, Finland. He coauthored more than 600 publications and supervised 40 Ph.D. dissertations. He is internationally recognized for his research in the areas of nonlinear signal and image processing and analysis. In addition, his research interests include multimedia analysis, indexing and retrieval, machine learning, voice conversion, and video processing and coding.

Dr. Gabbouj is a Member of the Academia European and the Finnish Academy of Science and Letters. He is a Member of the IEEE Fourier Award, and the Past-Chairman of the DSP Technical Committee of the IEEE Circuits and Systems Society. He was an Honorary Guest Professor of Jilin University, China, during 2005–2010. He served as a Distinguished Lecturer for the IEEE Circuits and Systems Society in 2004–2005, and the Past-Chairman of the IEEE-EURASIP Nonlinear Signal and Image Processing Board. He was the Chairman of the Algorithm Group of the EC COST 211quat. He served as an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, and was the Guest Editor of the *Multimedia Tools and Applications*, the *European Journal Applied Signal Processing*. He is the Past-Chairman of the IEEE Finland Section and the IEEE SP/CAS Finland Chapter. He was also the Chairman and Technical Program Chair of many national and international conferences and workshops. He is a member of IEEE SP and CAS Societies. He received the 2012 Nokia Foundation Visiting Professor Award, the 2005 Nokia Foundation Recognition Award, and many best paper awards.

QUERIES

Q1. Author: Please provide the year in Ref. [22].

Q2. Author: Please provide the year in which the author "Moncef Gabbouj" became a Fellow of the IEEE.

Real-Time Patient-Specific ECG Classification by 1-D Convolutional Neural Networks

Serkan Kiranyaz*, Turker Ince, and Moncef Gabbouj, *Fellow, IEEE*

***Abstract*—*Goal:* This paper presents a fast and accurate patient-specific electrocardiogram (ECG) classification and monitoring system. *Methods:* An adaptive implementation of 1-D convolutional neural networks (CNNs) is inherently used to fuse the two major blocks of the ECG classification into a single learning body: feature extraction and classification. Therefore, for each patient, an individual and simple CNN will be trained by using relatively small common and patient-specific training data, and thus, such patient-specific feature extraction ability can further improve the classification performance. Since this also negates the necessity to extract hand-crafted manual features, once a dedicated CNN is trained for a particular patient, it can solely be used to classify possibly long ECG data stream in a fast and accurate manner or alternatively, such a solution can conveniently be used for real-time ECG monitoring and early alert system on a light-weight wearable device. *Results:* The results over the MIT-BIH arrhythmia benchmark database demonstrate that the proposed solution achieves a superior classification performance than most of the state-of-the-art methods for the detection of ventricular ectopic beats and supraventricular ectopic beats. *Conclusion:* Besides the speed and computational efficiency achieved, once a dedicated CNN is trained for an individual patient, it can solely be used to classify his/her long ECG records such as Holter registers in a fast and accurate manner. *Significance:* Due to its simple and parameter invariant nature, the proposed system is highly generic, and, thus, applicable to any ECG dataset.**

***Index Terms*—Convolutional neural networks (CNNs), patient-specific ECG classification, real-time heart monitoring.**

I. Introduction

DESPITE the easiness of acquiring the data, there are still challenges ahead of us in order to extract reliable information from biomedical signals. Each heartbeat in the cardiac cycle shows the time evolution of the heart's electrical activity, which is made up of distinct electrical depolarization-repolarization patterns of the heart. For an expert cardiologist, any anomaly over the heart rate or rhythm or change in the morphological pattern over a recorded ECG waveform can easily be detected as an indication of an arrhythmia. However, this can turn out to be very challenging task for an automatic computerized system due to several reasons. Certain contaminations of biomedical signals to physiological artefact and external noise as well as imbalanced classes among biomedical signals (e.g., N- and S-type beats in an ECG signal) make the system's performance and accuracy significantly varying from patient to patient. Particularly, the time-varying dynamics and the morphological characteristics of ECG signals show significant variations for different patients and under different temporal and physical conditions. Even for the ECG of a healthy subject, which appears to be deterministic, the shapes of QRS complex, P waves, and R–R intervals will not be the same from one beat to the other under different circumstances [1].

There have been several methods for generic and fully automatic ECG classification based on signal processing techniques, such as frequency analysis [2], wavelet transform [3] and filter banks [4], statistical [5] and heuristic approaches [6], hidden Markov models [7], support vector machines [8], artificial neural networks (ANNs) [9], and mixture-of-experts method [10]. Generally speaking, they have not performed well in practice due to the aforementioned interpatient variations of the ECG signals, and thus, they usually exhibit a common drawback of having an inconsistent performance when, for instance, classifying a new patient's ECG signal. This makes them unreliable to be widely used clinically or in practice, and they tend to have high variations in their accuracy and efficiency for larger databases [11], [12].

Another severe problem is the lack of application of the common practice when evaluating and testing a particular method over a benchmark dataset. For this purpose, the *Association for the Advancement of Medical Instrumentation* (AAMI) provides standards and recommended practices for performance results of automated arrhythmia detection algorithms [13]. However, among many methods in the literature, only few [10], [14]–[18] have in fact used the AAMI standards along with the complete data from the benchmark MIT-BIH arrhythmia database [22]. Among all, only few of them with a patient-specific design [10], [12], [15]–[18] have, in particular, demonstrated significant performance improvements over the automatic and generic ECG classification methods thanks to their ability to adapt or optimize the classifier body according to each patient's ECG signal.

The aforementioned patient-specific ECG classification systems have a common approach with two major operations: feature extraction and training classification over the extracted features. They demonstrated that the ECG classification performance strongly depends on the characterization power of the features extracted from the ECG data. In the ECG classification literature, a vast number of features, their combinations, and feature selection approaches have been proposed [20]. In a former work, *Hermite* transform coefficients [15] achieved such a performance that is significantly higher than the others.

Manuscript received March 13, 2015; revised June 7, 2015, June 22, 2015, and July 22, 2015; accepted August 7, 2015. Date of publication; date of current version. *Asterisk indicates corresponding author.*

*S. Kiranyaz is with Electrical Engineering Department, College of Engineering, Qatar University, Doha, Qatar (e-mail: mkiranyaz@qu.edu.qa).

T. Ince is with the Izmir University of Economics

M. Gabbouj is with the Tampere University of Technology

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

Digital Object Identifier 10.1109/TBME.2015.2468589

Due to its time–frequency localization properties, the wavelet transform proves to be an efficient tool for analyzing nonstationary ECG signals [19]. In our prior work, Kiranyaz *et al.* [16], [17] that achieved superior performance than [15], we used translation-invariant dyadic wavelet transform to extract morphological features, and in order to avoid the well-known "Curse of Dimensionality" phenomenon and to significantly reduce redundancies in such a high-dimensional data space, the dimension of the input feature vectors has further been reduced by using principal component analysis (PCA). The lower dimensional morphological feature vector was then combined with two critical temporal features to form the final feature vector. However, using such fixed and hand-crafted features may not represent the characteristics of the underlying signal in an optimal way and obviously this is against the philosophy of a "patient-specific" approach since the same set of features will be used for all patients under all circumstances. The true "patient-specific" solution indeed requires the design of the best possible features for each individual ECG data. Moreover, extracting several features, especially in the transform domains along with the postprocessing methods such as PCA may significantly increase the computational complexity of the overall process, and this may hinder them from the usage in light-weight applications (e.g., mobile or wearable health monitoring devices) or for the classification of large ECG records such as Holter registers.

In order to address such deficiencies and drawbacks, in this paper, we propose a novel ECG classification approach based on adaptive 1-D convolutional neural networks (CNNs). CNNs are hierarchical neural networks whose convolutional layers alternate with subsampling layers, reminiscent of simple and complex cells in the human visual cortex [21], following with a fully connected layers, which are identical to multilayer perceptrons (MLP). They primarily mimic the human visual system, which can efficiently recognize the patterns and structures (e.g., objects) in a visual scenery. CNNs are now commonly used for the "deep learning" tasks, such as object recognition in large image achieves while achieving the state-of-the-art performances [24]–[26]. To our knowledge, this is the first study, where they are used over 1-D signals, in particular, for the purpose of ECG classification and anomaly detection. With the proposed adaptation over the traditional CNNs, the proposed approach can classify each heart beat with any sampling rate; therefore, voiding the need for any manual feature extraction and postprocessing. With the proper training, the convolutional layers of CNNs can learn to extract patient-specific features, while the MLP layers perform the classification task to produce the final class vectors of each beat. With the limited training data as proposed in [10] and [14]–[17], we shall demonstrate that simple CNNs will suffice to achieve a superior classification performance rather than the complex ones that are commonly used for deep learning tasks. As a result, simple 1-D CNNs are easier to train with only few dozens of back-propagation (BP) epochs, and can, thus, perform the classification task with utmost speed (requiring only few hundreds of 1-D convolutions). This makes them a perfect choice for real-time ECG monitoring and early alert system on light-weight devices. An illustration of the proposed approach is shown in Fig. 1. Finally, we aim to achieve a high level of *robustness* with respect to the variations of the dataset, since the proposed system is designed with a minimum set of parameters and manual settings thanks to the combined learner for feature extraction and classification.

The rest of this paper is organized as follows. Section II outlines the ECG dataset used in this study, and provides a detailed description of the possible raw data representations for the proposed patient-specific heartbeat classification system. The adaptive 1-D CNNs along with the BP training method are presented in Section III. In Section IV, the performance and robustness of the proposed approach are evaluated over the MIT/BIH arrhythmia database using the standard performance metrics, and the results are compared with the previous state-of-the-art works. Finally, Section V concludes this paper.

II. ECG Data Processing

In this study, ECG datasets from the MIT/BIH arrhythmia database [22] are used for the performance evaluation of the proposed patient-specific ECG approach. This benchmark database contains 48 records, each containing two-channel ECG signals for 30-min duration selected from 24-h recordings of 47 individuals. Continuous ECG signals are bandpass filtered at 0.1–100 Hz and then digitized at 360 Hz. The database contains annotation for both timing information and beat class information verified by independent experts. In this study, we followed the identical data partitioning as in [16] and [17] so as to comply with the AAMI ECAR-1987 recommended practice [13]. We used 44 records from the MIT/BIH arrhythmia database, excluding four records, which contain paced heartbeats. The first 20 records (numbered in the range of 100–124), which include representative samples of routine clinical recordings, are used to select representative beats to be included in the common training data. The remaining 24 records (numbered in the range of 200–234) contain uncommon but clinically significant arrhythmias, such as ventricular, junctional, and supraventricular arrhythmias [27]. A total of 83648 beats from all 44 records are used as test patterns for performance evaluation. AAMI recommends that each ECG beat be classified into the following five heartbeat types: N (beats originating in the sinus mode), S (supraventricular ectopic beats), V (ventricular ectopic beats), F (fusion beats), and Q (unclassifiable beats). For all records, we used the modified-lead II signals, and utilized the labels to locate beats in ECG data. The beat detection process is beyond the scope of this paper, as many highly accurate ($> 99\%$) beat detection algorithms have been reported in the literature [19], [23].

The raw data of each beat are represented by 64 or 128 samples by downsampling where the latter is intended for the evaluation of the higher resolution data representation. As illustrated in Fig. 1, in order to learn the morphological structure of the beat, equal number of samples from each side from the R (center) point of the beat are fed into a neuron of the CNN's input layer. In order to learn the temporal characteristics of each beat, a beat trio is formed from its neighbor beats, and is fed into another neuron at the input layer. Therefore, the difference in timing information of the center beat together with its

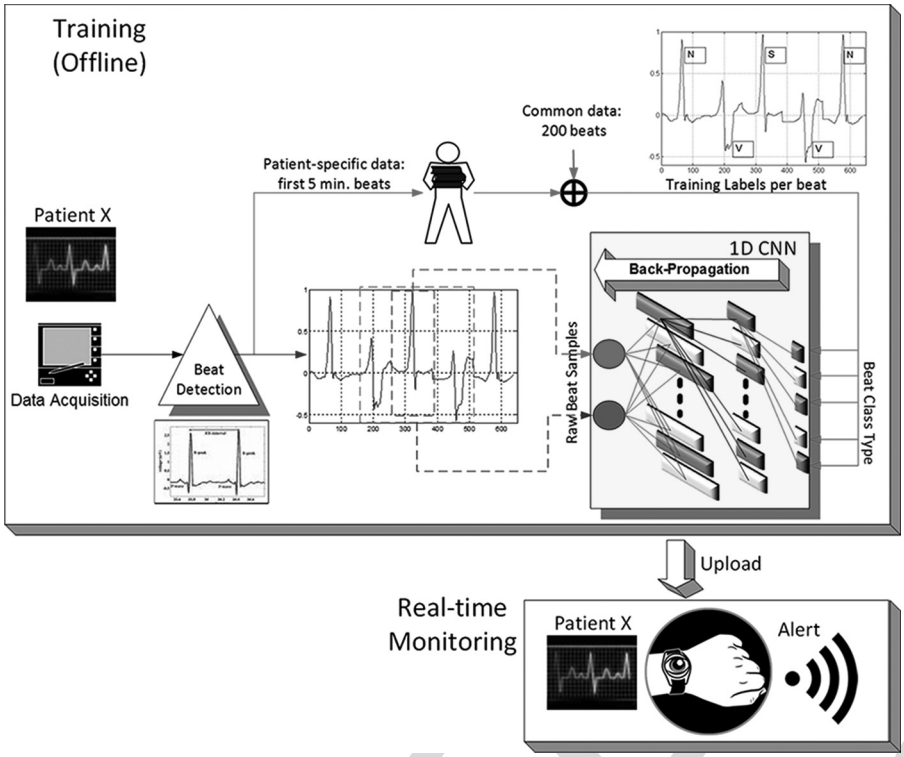

Fig. 1. Overview of the proposed approach in training (offline) and real-time classification and monitoring phases.

neighbors in the beat-trio formation can indicate timing information related ECG anomalies such as the presence of an APC (S) beat. This is the *base* representation of each beat's raw data and on top of this, FFT of each beat (both magnitude and phase) will also be considered as the *extended* raw data representation in the frequency domain. The purpose is to evaluate the performance gain—if any—obtained by such extension in raw-data representation.

The data used for training the individual patient's classifier consist of two parts: global (common to each patient) and local (patient-specific) training patterns. While patient-specific data contain the first 5-min segment of each patient's ECG record and is used as a part of the training data to perform patient adaptation, the global dataset contains a relatively small number of representative beats from each class in the training files, and helps the classifier learn other arrhythmia patterns that are not included in the patient-specific data. This practice conforms to the AAMI recommended procedure allowing the usage of at most 5-min section from the beginning of each patient's recording for training [13].

III. Adaptive 1-D CNNs

As mentioned earlier, adaptive 1-D CNNs are used for both feature extraction and classification of the raw ECG data from each individual patient in the database. In Appendix A, we introduced an overview of the traditional CNNs developed for a 2-D image classification. Accordingly, we shall present the design of our adaptive CNNs in accordance with the traditional CNNs in 2-D and formulate its BP training. Finally, we shall highlight the changes and modifications needed for 1-D CNNs from their 2-D counterparts along with the BP formulations.

To simplify the CNN analogy and to have the freedom of any input layer dimension independent from the CNN parameters, the neurons of the hidden CNN layers are extended such that they are capable of both convolution and downsampling as shown in Fig. 2. This implementation also allows the ability of a "CNN-only" design without the MLP layers. For the illustration purpose, we assume 3×3 kernels ($Kx = Ky = 3$) for all CNN layers in the figure; however, different kernel sizes can also be assigned if desired. The final output of the *k*th neuron at layer *l*, s_k^l, is, therefore, the subsampled version of the intermediate output y_k^l. During the forward propagation (FP), the input map of the next layer neuron will be obtained by the cumulation of the final output maps of the previous layer neurons convolved with their individual kernels as follows:

$$x_k^l = b_k^l + \sum_{i=1}^{N_{l-1}} \text{conv2D}(w_{ik}^{l-1}, s_i^{l-1}) \quad (1)$$

where conv2D (.,.) is a regular 2-D convolution without zero padding on the boundaries, x_k^l is the input, b_k^l is the bias of the *k*th neuron at layer *l*, and s_i^{l-1} is the output of the *i*th neuron at layer *l*–1. w_{ik}^{l-1} is the kernel (weight) from the *i*th neuron at layer *l*–1 to the *k*th neuron at layer *l*. To accomplish BP training, there are three more elements that are stored for each neuron: the delta error Δ_k^l, downsampled delta error Δ_{sk}^l, and, finally, the derivative of the intermediate output $f'(x_k^l)$, all of which will be explained in the next section.

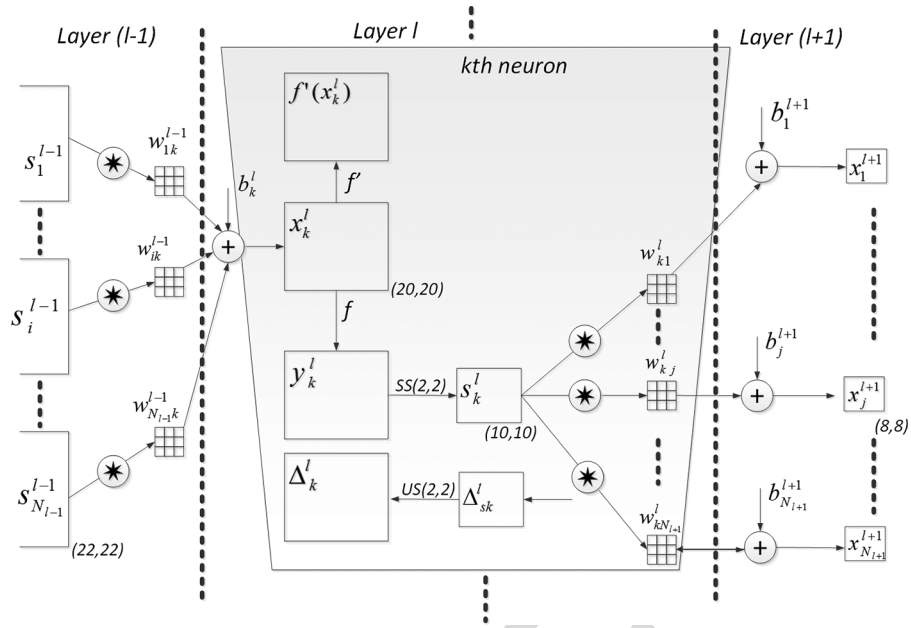

Fig. 2. Adaptive CNN implementation.

We aim that the number of hidden CNN layers can be set to any number. This ability is possible in this implementation because the subsampling factor of the output CNN layer (the hidden CNN layer just before the first MLP layer) is automatically set to the dimensions of its input map, e.g., in Fig. 2, if the layer $l+1$ would be the output CNN layer, then the subsampling factors for that layer will be $ssx = ssy = 8$ since the input map dimension is 8×8 in this sample illustration. Besides the subsampling, note that the dimension of the input maps will gradually decrease due to the convolution without zero padding, i.e., in Fig. 2, the dimension of the neuron output is 22×22 at the layer $l-1$ that is reduced to 20×20 at the layer l. As a result of this, the dimension of the input maps of the current layer is reduced by (Kx–1, Ky–1), where Kx and Ky are the width and height of the kernel, respectively.

A. Intra-BP Within a CNN Neuron: $\Delta_k^l \longleftarrow \Delta s_k^l$

The BP among MLP layers and from the first MLP layer to the output CNN layer is covered in Appendix. Once the first BP is performed from the next layer, $l+1$, to the current layer, l, then we can further BP it to the input delta. Let zero order upsampled map be: $us_k^l = up_{ssx,ssy}(s_k^l)$, then one can write

$$\begin{aligned}\Delta_k^l = \frac{\partial E}{\partial x_k^l} &= \frac{\partial E}{\partial y_k^l}\frac{\partial y_k^l}{\partial x_k^l} = \frac{\partial E}{\partial us_k^l}\frac{\partial us_k^l}{\partial y_k^l} f'(x_k^l) \\ &= up(\Delta s_k^l)\, \beta\, f'(x_k^l) \end{aligned} \quad (2)$$

where $\beta = (ssx.ssy)^{-1}$ since each pixel of s_k^l was obtained by averaging $ssx.ssy$ number of pixels of the intermediate output y_k^l. If maximum pooling is used instead of averaging, then (2) should be adapted accordingly.

B. Inter-BP Among CNN Layers: $\Delta s_k^l \xleftarrow{\Sigma} \Delta_l^{l+1}$

Recall the basic rule of BP: if the output of the kth neuron at layer l, contributes a neuron i in the next level with a weight, w_{ki}^l, that next layer neuron's delta, Δ_i^{l+1}, will contribute with the same weight to form Δ_k^l of the neuron in the previous layer l. This means

$$\frac{\partial E}{\partial s_k^l} = \Delta s_k^l \xleftarrow{\Sigma} \Delta_i^{l+1}\ \forall i \in \{1, N_{l+1}\} \quad (3)$$

Specifically

$$\frac{\partial E}{\partial s_k^l} = \Delta s_k^l = \sum_{i=1}^{N_{l+1}} \frac{\partial E}{\partial x_i^{l+1}}\frac{\partial x_i^{l+1}}{\partial s_k^l} = \sum_{i=1}^{N_{l+1}} \Delta_i^{l+1}\frac{\partial x_i^{l+1}}{\partial s_k^l} \quad (4)$$

where

$$x_i^{l+1} = \ldots + s_k^l * w_{ki}^l + \cdots \quad (5)$$

where "$*$" is the regular conv2D(.,.) operator without zero padding. It is obviously hard to compute the derivative directly from the 2-D convolution. Instead let us focus on the contribution of a single output pixel, $s_k^l(m,n)$, to the pixels of the next layer map, $x_i^{l+1}(m,n)$, assuming a 3×3 kernel

$$\begin{aligned} &x_i^{l+1}(m-1,n-1) = \ldots + s_k^l(m,n).w_{ki}^l(2,2) + \ldots \\ &x_i^{l+1}(m-1,n) = \ldots + s_k^l(m,n).w_{ki}^l(2,1) + \ldots \\ &\ldots\ldots \\ &x_i^{l+1}(m+1,n+1) = \ldots + s_k^l(m,n).w_{ki}^l(0,0) + \ldots \end{aligned} \quad (6)$$

This is illustrated in Fig. 3, where the role of an output pixel, $s_k^l(m,n)$, over two pixels of the next layer's input neuron's pixels $x_i^{l+1}(m-1,n-1)$and $x_i^{l+1}(m+1,n+1)$ can be seen.

Considering the pixel as a MLP neuron connected to other MLP neurons in the next layer, according to the basic rule of

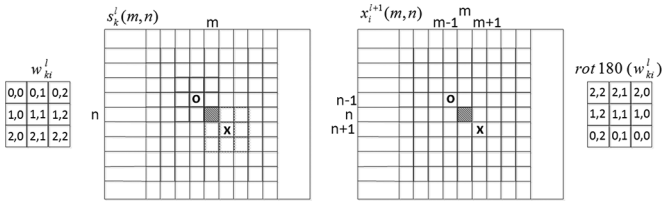

Fig. 3. Single pixel's contribution of the output $s_k^l(m,n)$ to the two pixels of the x_i^{l+1} using a 3×3 kernel.

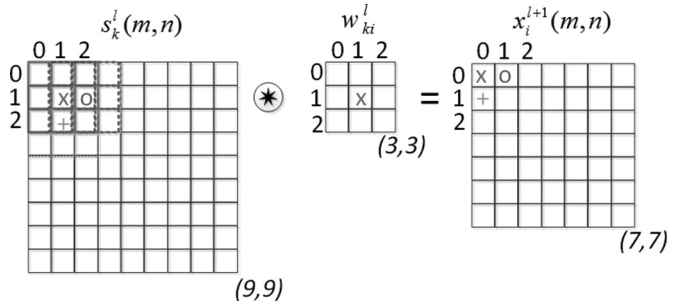

Fig. 4. Convolution of the output of the current layer neuron s_k^l and kernel w_{ki}^l to form the input of the *i*th neuron x_i^{l+1} at the next layer $l+1$.

BP one can then easily write the delta of $s_k^l(m,n)$ as

$$\frac{\partial E}{\partial s_k^l}(m,n) = \Delta s_k^l(m,n) = \sum_{i=1}^{N_{l+1}} \left(\sum_{r=-1}^{1} \sum_{t=-1}^{1} \Delta_i^{l+1}(m+r,n+t) \cdot w_{ki}^l(1-r,1-t) \right) \quad (7)$$

generalizing it for all pixels of the Δs_k^l yields

$$\Delta s_k^l = \sum_{i=1}^{N_{l+1}} \text{conv2D}z\left(\Delta_i^{l+1}, \text{rot180}(w_{ki}^l)\right) \quad (8)$$

where rot180(.) rotates the kernel, w_{ki}^l, 180°, and then conv2D*z*(.,.) performs a *full* convolution with zero padding by (*Kx*–1, *Ky*–1) zeros to each boundary of the Δ_i^{l+1} in order to achieve equal dimensions (width and height) for Δs_k^l and Δ_i^{l+1} with the s_k^l.

C. Computation of the Weight (Kernel) and Bias Sensitivities

As in the regular BP on MLPs, the delta of the *i*th neuron at layer $l+1$, Δ_i^{l+1} will be used to update the bias of that neuron, and all weights of the neurons in the previous layer connected to that neuron as given in (19):

$$x_i^{l+1} = b_i^{l+1} + \cdots + y_k^l w_{ki}^l + \cdots$$
$$\therefore \frac{\partial E}{\partial w_{ki}^l} = y_k^l \ \Delta_i^{l+1} \text{ and } \frac{\partial E}{\partial b_i^{l+1}} = \Delta_i^{l+1}. \quad (9)$$

Recall the update rule: The sensitivity of the weight connecting the *k*th neuron in the current layer to the *i*th neuron in the next layer depends on the output of the current layer neuron, and the delta of the next layer neuron. For CNN layer neurons, we need to follow a similar approach to find out weight and bias sensitivities. Fig. 4 illustrates the convolution of the output of the current layer neuron s_k^l and kernel w_{ki}^l to form the input of the *i*th neuron x_i^{l+1} at the next layer $l+1$.

So now, we can focus on the contribution of each kernel element over the output. The following expressions can be written for the sample 2-D convolution shown in Fig. 4:

$$\begin{aligned}
x_i^{l+1}(0,0) &= \cdots + w_{ki}^l(0,0)s_k^l(0,0) + w_{ki}^l(0,1)s_k^l(0,1) \\
&\quad + w_{ki}^l(1,0)s_k^l(1,0) + \cdots \\
x_i^{l+1}(0,1) &= \cdots + w_{ki}^l(0,0)s_k^l(0,1) + w_{ki}^l(0,1)s_k^l(0,2) \\
&\quad + w_{ki}^l(1,0)s_k^l(1,1) + \cdots \\
x_i^{l+1}(1,0) &= \cdots + w_{ki}^l(0,0)s_k^l(1,0) + w_{ki}^l(0,1)s_k^l(1,1) \\
&\quad + w_{ki}^l(1,0)s_k^l(2,0) + \cdots \\
x_i^{l+1}(m,n) &= \cdots + w_{ki}^l(0,0)s_k^l(m,n) + w_{ki}^l(0,1) \\
&\quad \times s_k^l(m,n+1) + w_{ki}^l(1,0)s_k^l(m+1,n) + \cdots \\
x_i^{l+1}(m,n) &= \sum_{r=-1}^{1} \sum_{t=-1}^{1} w_{ki}^l(r+1,t+1) \\
&\quad \times s_k^l(m+r+1,n+t+1). \quad (10)
\end{aligned}$$

Since each weight (kernel) element is used (shared) in common to form each neuron input, $x_i^{l+1}(m,n)$, the derivative will be the cumulation of delta-output product for all pixels, i.e.,

$$\frac{\partial E}{\partial w_{ki}^l(r,t)} = \sum_m \sum_n \Delta_i^{l+1}(m,n) s_k^l(m+r,n+t)$$
$$\Rightarrow \frac{\partial E}{\partial w_{ki}^l} = \text{conv2D}(s_k^l, \Delta_i^{l+1}). \quad (11)$$

Similarly, the bias of this neuron, b_k^l, contributes to all pixels in the image (same bias shared among all pixels), so its sensitivity will be the cumulation of individual pixel sensitivities i.e.,

$$\frac{\partial E}{\partial b_k^l} = \sum_m \sum_n \frac{\partial E}{\partial x_k^l(m,n)} \frac{\partial x_k^l(m,n)}{\partial b_k^l} = \sum_m \sum_n \Delta_k^l(m,n). \quad (12)$$

As a result, the iterative flow of the BP can be stated as follows.

1) Initialize weights [usually randomly, U(–a, a)].
2) For each BP iteration DO:
   a) For each item (or a group of items or all items) in the dataset, DO:

i) *FP:* FP from the input layer to the output layer to find outputs of each neuron at each layer $y_i^l \; \forall i \in [1, N_l]$ and $\forall l \in [1, L]$.
ii) *BP:* Compute delta error at the output layer and BP it to first hidden layer to compute the delta errors $\Delta_k^l \; \forall k \in [1, N_l]$ and $\forall l \in [2, L-1]$.
iii) *PP:* Postprocess to compute the weight and bias sensitivities using (11), (12), and (25).
iv) *Update:* Update the weights and biases with the (cumulation of) sensitivities found in (c) scaled with the learning factor ε:

$$w_{ik}^{l-1}(t+1) = w_{ik}^{l-1}(t) - \varepsilon \frac{\partial E}{\partial w_{ik}^{l-1}}$$
$$b_k^l(t+1) = b_k^l(t) - \varepsilon \frac{\partial E}{\partial b_k^l} \qquad (13)$$

B. Changes for the Adaptive 1-D CNN Implementation

There are only some minor differences between the 2-D and 1-D CNNs. The main difference is obviously the 1-D arrays used in each neuron for its kernels (weights), input and output elements for both FP and BP rather than 2-D matrices. Therefore, during both FP and BP runs, 1-D array manipulations such as conv1D and reverse will be performed instead of 2-D matrix operations, such as conv2D and rot180. The 2-D parameters for kernel size (*Kx, Ky*) and subsampling (*ssx, ssy*) are now single scalars *K* and *ss*, respectively. If present, the MLP layers of the 1-D CNN are identical to the 2-D counterpart, and, therefore, same equations can be used for both FP and BP on those layers. The expression of the 2-D FP given in (26) can now be written as

$$x_k^l = b_k^l + \sum_{i=1}^{N_{l-1}} \text{conv1}D(w_{ik}^{l-1}, s_i^{l-1}). \qquad (14)$$

The inter-BP delta error of the output s_k^l given in (8) can now be expressed for 1-D as

$$\Delta s_k^l = \sum_{i=1}^{N_{l+1}} \text{conv1}Dz\left(\Delta_i^{l+1}, \text{rev}(w_{ki}^l)\right) \qquad (15)$$

where rev(.) reverses the array, and conv1*Dz*(.,.) performs full convolution in 1-D with *K*–1 zero padding. The intra-BP is also identical to the one for 2-D CNNs, as expressed in (2), while $\beta = ss^{-1}$ and a 1-D upsampling operator is now used. Finally, the weight and bias sensitivities given in (11) and (12) for 2-D can now be expressed for 1-D CNNs as

$$\frac{\partial E}{\partial w_{ki}^l} = \text{conv1D}(s_k^l, \Delta_i^{l+1})$$
$$\frac{\partial E}{\partial b_k^l} = \sum_n \Delta_k^l(n). \qquad (16)$$

IV. Experimental Results

In this section, we shall first present the experimental setup for the test and evaluation of the proposed patient-specific ECG classification approach. We shall then present the overall results obtained from the ECG classification experiments and perform comparative evaluations against several state-of-the-art techniques in this field. The robustness of the proposed system against variations of signal resolution (hence, the CNN configuration) and raw data representation will then be evaluated. Finally, the computational complexity of the proposed method for both training and classification will be evaluated in detail.

A. Experimental Setup

As mentioned earlier, in order to evaluate the effects of different resolutions and raw data representations particularly in the transform domain, we used two alternatives for each case. The beats are represented in both 64 and 128 samples centered on the R-peak, and we used the FFT representation (magnitude and phase) of each beat as in the extended data representation. We purposefully used a simple 1-D CNN in all experiments with only three CNN layers and two MLP layers, in order to achieve the utmost computational efficiency for both training and particularly for real-time classification. On top of this, we aim to demonstrate that deep learners are not indeed needed to achieve a superior ECG classification performance. The 1-D CNN used in all experiments has 32 and 16 neurons on the first- and second-hidden CNN layers and ten neurons on the hidden MLP layer. The output (MLP) layer size is 5 which is the number of beat classes and the input (CNN) layer size is either 2 (base*)* or 4 (extended*)* according to the choice of raw data representation. For 64 and 128 sample beat representations, the kernel sizes are set to 9 and 15, and the subsampling factors are set to 4 and 6, respectively. As a result, in the proposed adaptive CNN implementation, the subsampling factors for the last CNN layers are automatically set to 6 and 5, respectively.

For all experiments, we employ a shallow training: the maximum number of BP iterations is set to 50, and another stopping criterion is the minimum train classification error level that is set to 3% to prevent overfitting. Therefore, the training will terminate if either of the criteria is met. We initially set the learning factor ε as 0.001, and applied a global adaptation during each BP iteration; if the train mean-squared error (MSE) decreases in the current iteration, we slightly increase ε by 5%; otherwise, we reduce it by 30% for the next iteration. As BP is a deterministic gradient-descent optimization technique, which makes it quite dependent to the initial (random) setting of the network parameters (kernels, weights, and biases), we performed ten individual BP runs for each patient in the database, and the average classification performance is reported.

B. Base Classification Performance Evaluation

We performed classification experiments on 44 records of the MIT/BIH arrhythmia database, containing a total of 100389 beats to be classified into five heartbeat types according to the AAMI recommendation [13]. For training the 1-D CNNs, both common and patient-specific training patterns are used; the common part of the training dataset contains a total of 245 representative beats, including 75 from each type-N, type-S, and type-V beats, and all (13) type-F and (7) type-Q beats, randomly

TABLE I
CONFUSION MATRICES OF THE ECG BEAT CLASSIFICATION RESULTS FOR ALL 44 RECORDS (BOTTOM) AND FOR THE 24 TEST RECORDS (TOP) IN THE MIT/BIH ARRHYTHMIA DATABASE

		Classification Result				
Ground Truth		N	S	V	F	Q
	N	40963F (40532)	807 (776)	350 (382)	67 (56)	4 (20)
	S	625 (672)	1440 (1441)	149 (197)	14 (5)	1 (5)
	V	114 (392)	69 (299)	4247 (4022)	39 (75)	2 (32)
	F	82 (164)	4 (26)	70 (46)	497 (378)	0 (2)
	Q	6 (6)	2 (0)	5 (1)	0 (1)	0 (0)

		Classification Result				
Ground Truth		N	S	V	F	Q
	N	73539 (73019)	824 (991)	368 (513)	69 (98)	5 (29)
	S	837 (686)	1568 (1568)	178 (205)	15 (5)	2 (6)
	V	230 (462)	72 (333)	5277 (4993)	39 (79)	4 (32)
	F	92 (168)	4 (28)	73 (48)	503 (379)	0 (2)
	Q	31 (8)	2 (1)	5 (3)	0 (1)	4 (1)

The previous results from [16] are shown in parentheses.

TABLE II
VEB AND SVEB CLASSIFICATION PERFORMANCE OF THE PROPOSED METHOD AND COMPARISON WITH FOUR MAJOR ALGORITHMS FROM THE LITERATURE

Methods	VEB				SVEB			
	Acc	*Sen*	*Spe*	*Ppr*	*Acc*	*Sen*	*Spe*	*Ppr*
Hu *et al.* [10][1]	94.8	78.9	96.8	75.8	N/A	N/A	N/A	N/A
Jiang and Kong [15] [1]	98.8	94.3	99.4	95.8	**97.5**	74.9	98.8	78.8
Ince *et al.* [16][1]	97.9	90.3	98.8	92.2	96.1	**81.8**	98.5	63.4
Proposed[1]	**98.9**	**95.9**	**99.4**	**96.2**	96.4	68.8	**99.5**	**79.2**
Jiang and Kong [15] [2]	98.1	86.6	**99.3**	**93.3**	**96.6**	50.6	**98.8**	**67.9**
Ince *et al.* [16][2]	97.6	83.4	98.1	87.4	96.1	62.1	98.5	56.7
Proposed[2]	**98.6**	**95**	98.1	89.5	96.4	**64.6**	98.6	62.1
Ince *et al.* [16] [3]	98.3	84.6	98.7	87.4	97.4	**63.5**	99.0	53.7
Proposed[3]	**99**	**93.9**	**98.9**	**90.6**	**97.6**	60.3	**99.2**	**63.5**

Best results are highlighted.
[1]The comparison results are based on 11 common recordings for VEB detection and 14 common recordings for SVEB detection.
[2]The VEB and SVEB detection results are compared for 24 common testing records only.
[3]The VEB and SVEB detection results of the proposed system for all training and testing records.

sampled from each class from the first 20 records (picked from the range 100 to 124) of the MIT/BIH database, and the patient-specific training data include the beats from the first 5 min of the corresponding patient's ECG record. Patient-specific 1-D CNN networks are trained with a total of 245 common training beats, and a variable number of patient-specific beats depending on the patient's heart rate, so only less than 1% of the total beats are used for training. The remaining beats (25 min) of each record, in which 24 out of 44 records are completely new to the classifier, are used as test patterns for performance evaluation.

Classification performance is measured using the four standard metrics found in the literature [10]: classification accuracy (*Acc*), sensitivity (*Sen*), specificity (*Spe*), and positive predictivity (*Ppr*). While accuracy measures the overall system performance over all classes of beats, the other metrics are specific to each class, and they measure the ability of the classification algorithm to distinguish certain events (i.e., VEBs or SVEBs) from nonevents (i.e., non-VEBs or non-SVEBs). The respective definitions of these four common metrics using true positive (*TP*), true negative (*TN*), false positive (*FP*), and false negative (*FN*) are as follows: *Accuracy* is the ratio of the number of correctly classified patterns to the total number of patterns classified, $Acc = (TP+TN)/(TP+TN+FP+FN)$; *Sensitivity* is the rate of correctly classified events among all events, $Sen = TP/(TP+FN)$; *Specificity* is the rate of correctly classified nonevents among all nonevents, $Spe = TN/(TN+FP)$; and *Positive Predictivity* is the rate of correctly classified events in all detected events, $Ppr = TP/(TP+FP)$. Since there is a large variation in the number of beats from different classes in the training/testing data (i.e., 39465/50354 type-N, 1277/5716 type-V, and 190/2571 type-S beats), sensitivity, specificity, and positive predictivity are more relevant performance criteria for medical diagnosis applications.

TABLE III
CONFUSION MATRIX FOR THE ECG BEAT CLASSIFICATION OF THE PATIENT 202

		Classification Result				
Ground Truth		N	S	V	F	Q
	N	1435	258	10	0	0
	S	11	15	28	0	0
	V	4	1	10	0	0
	F	1	0	0	0	0
	Q	0	0	0	0	0

For the base performance evaluation, we shall consider the *base* raw data represented by 128 samples (both the single beat and the beat trio). The results of the other three alternatives from the extended raw data and 64 samples representation over the other 1-D CNN setup will be considered to demonstrate the robustness of the system against the variations on raw data representations, signal resolution, and CNN configurations. For the proposed approach and the previous state-of-the-art method in [16], Table I presents the confusion matrices of ECG beat classification results for all 44 records and the 24 records of the test dataset. To perform a more extensive and accurate comparative performance evaluation, the base performance of the proposed system is compared with the three existing algorithms, [10], [15], and [16], all of which comply with the AAMI standards. Although P. de Chazal in [12] and [14] also comply with the AAMI standards, since they used different training and test data, it is not possible to directly compare their results with the proposed method. In accordance with the AAMI recommendations, the problem of VEB and SVEB detection is considered individually and results are summarized in Table II. The benchmark MIT/BIH database is partitioned into three evaluation datasets. For VEB detection, the dataset 1 contains 11 test recordings (200, 202, 210, 213, 214, 219, 221, 228, 231, 233, and 234), and for SVEB detection, comparison results are based on 14 common recordings (with the addition of records 212, 222, and 232). Dataset 1 is common for all competing methods. Dataset 2 is the test partition of the benchmark database containing 24 records (200 and onward). Three methods (the proposed, [15], and [16]) are tested on this dataset. Finally, the dataset 3 is the

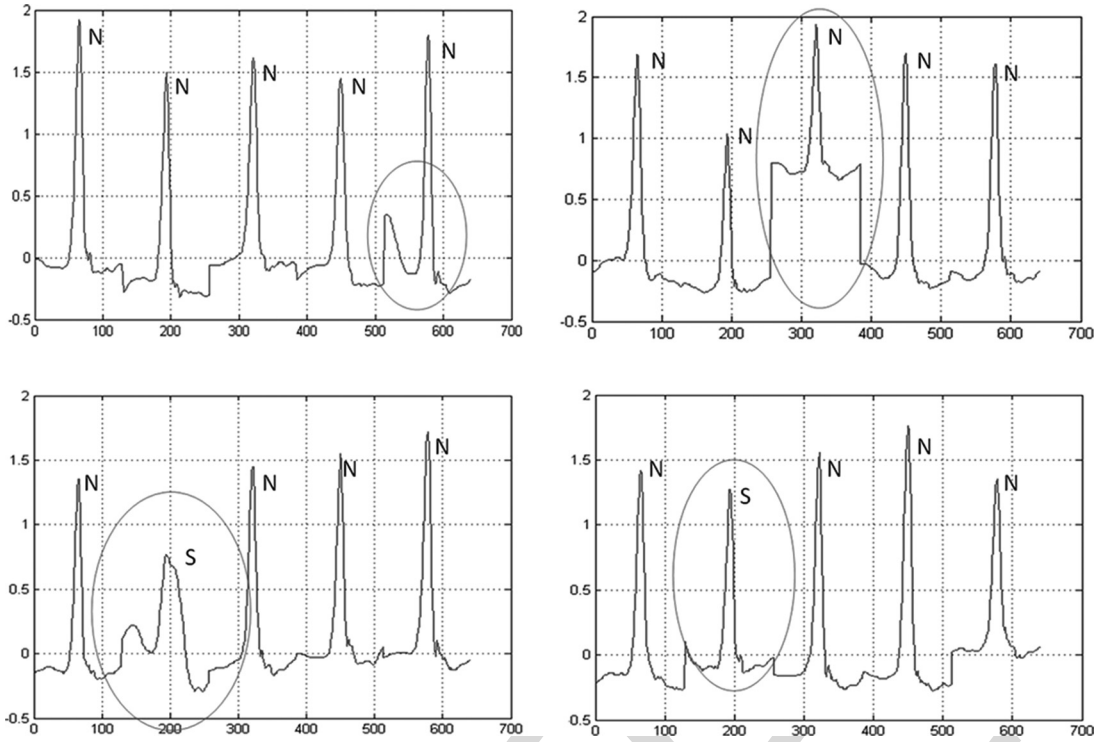

Fig. 5. Four 5-beats interval from the test section of the patient 202's ECG record with the ground truth labels.

entire database with all records over which two methods, the proposed and [16] are tested.

Several interesting observations can be made from the results in Table II. First, for SVEB detection, sensitivity and positive predictivity rates are comparably lower than VEB detection, while a high-specificity performance is achieved. The reason for the worse classifier performance in detecting SVEBs is that SVEB class is underrepresented in the training data, and, hence, more SVEB beats are misclassified as normal beats. Moreover, on several patients, particularly on the test dataset (dataset 2) both the patient specific data from the first 5-min interval and the common data of 200 beats extracted randomly from the training dataset do not successfully characterize most of the S beats and some of the V beats (e.g., patients 201, 202, 209, 222, and 232). Take, for instance, the confusion matrix given for patient 202 in Table III, where 258 normal beats are misclassified as S beats and 28 S beats are misclassified as V beats. The four 5-beat intervals from the test section of this patient's ECG record are shown in Fig. 5. Such anomalies shown in the plots on N and S beats do not in fact exist in the training dataset of this patient, and, hence, the classifier can easily misclassify them. Particularly, the plot in the bottom-left was misclassified as a V beat due to its morphological anomaly that was learned as a V beat, and the S beat in the bottom-right was misclassified as N beat due to its strong resemblance to the pattern of N beats. This is why, the selection of the common data is of utmost importance and rather than random, the selection should be performed with care to cover as much representative beats as possible to minimize such an inevitable source of classification errors.

In the next section, we shall test the robustness of the proposed approach against the variations of the signal resolution along with the network configuration and raw data representation to validate whether the performance can significantly deteriorate.

C. Robustness

Over each dataset partitioning, we evaluate the proposed approach by varying the beat resolution (64 versus 128 samples) along with the CNN parameters and also the raw data representation (base versus extended). We observed that mostly comparable performances with the base performance level are achieved for all measures on both VEB and SVEB classification, while the performance drop becomes somewhat significant ($\sim$8%) only for the positive predictivity of the SVEB classification. Other measures are usually slightly comparable with the base performance. It is quite evident that the effects of such variations over the classification accuracy are usually insignificant, especially for VEB classification. Therefore, other raw data, resolutions, and CNN parameter variants can be conveniently used within the proposed approach without sacrificing from the base performance level. The only cost using the base (128 samples per beat) setting rather than the 64 samples resolution is the increased computational complexity due to the larger 1-D convolutions during both BP and FP. This will be investigated in detail in next section.

D. Computational Complexity

We implemented the proposed adaptive 1-D CNN using C++ over MS Visual Studio 2013 in 64 bit. This is a non-GPU implementation; however, Intel OpenMP API is used to obtain multiprocessing with a shared memory. The experiments are performed on a computer with I7-4700MQ at 2.4 GHz

(eight CPUs) and 16-Gb memory. In theory, this should yield to 8× speed improvement, but, in practice, the observed speed improvement was between 4.8× and 5×.

Considering the baseline implementation (with base raw data with 128 samples beat resolution), the average time for 1 BP iteration per beat is about 4.24 m·sec with this setting. In a single CPU implementation, it becomes 21.2 m · sec. Considering a complete BP run with maximum 50 iterations over a patient's ECG training dataset with maximum 600 beats (maximum 400 patient specific + 200 common), this means that the maximum time for training would be $600 \times 50 \times 21.2$ m · sec = 10.6 min. The average training time for [16] was about 24 min, including the feature extraction and postprocessing operations. In our implementation with OpenMP, this was less than 2 min per patient, which is an insignificant time for training the system which will be performed only once per patient. For the 64 samples beat resolution, the average time for 1 BP iteration per beat drops down to 3.93 m · sec.

The most important advantage of the proposed system is its significantly low computational cost for the beat classification. Specifically, for the single-CPU implementation, the total time for a FP of a single beat to obtain the class vector is about 0.58 and 0.74 m · sec for 64 and 128 samples beat resolutions, respectively. Although there is now a significant decrease in the computational cost for using 64 samples beat resolution, note that this speed is still more than 1000× faster than the real-time requirement.

The main limitation of the proposed approach is that the characterization of the crucial anomaly beats such as the S beats. As discussed earlier, neither the patient specific data nor the common data that are randomly collected guarantees to represent such anomaly beats properly, and if they are not included in the training dataset, the system may entirely or partially fail to classify them. Furthermore, the proposed approach like the prior works in this domain do not support incremental or active learning, which hinders the ability to adapt to the changes of the patient's heartbeat patterns.

V. Conclusion

In this study, we proposed a patient-specific ECG heartbeat classifier with an adaptive implementation of 1-D CNNs that are able to fuse the two major blocks of the traditional ECG classification into a single learning body: feature extraction and classification. Such a compact implementation, for each patient over a simple CNN, not only negates the necessity to extract handcrafted manual features, or any kind of pre- and postprocessing, also makes it a primary choice for a real-time implementation for heart monitoring and anomaly detection. Besides the speed and computational efficiency achieved, the proposed method only requires 1-D convolutions (multiplications and additions) that make any hardware implementation simpler and cheaper. In addition to that, once a dedicated CNN is trained for an individual patient, it can solely be used to classify his/her long ECG records such as Holter registers in a fast and accurate manner.

The results of the classification experiments, which are performed over the benchmark MIT/BIH arrhythmia database show that for both test datasets (1 and 2), the base performance level of the proposed approach in both VEB and SVEB detection is comparable or better than the competing methods for most of the measures. Over the entire dataset (dataset 3), it has the highest performance measures except the SVEB sensitivity. We also observed that variations on the beat resolution with a different CNN settings and raw data representations are not degrading and can be desirable for even lower computational complexity. As a result, the proposed approach achieves the main design objectives, i.e., maintaining a fast, robust, and patient-specific system with a superior classification performance. As a future work, we are planning to design the hardware implementation of the proposed approach.

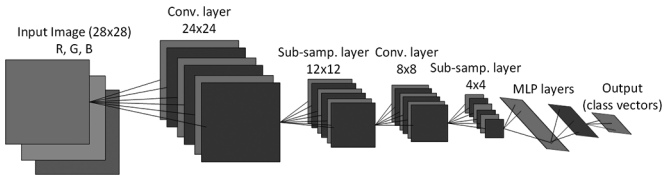

Fig. 6. Overview of a sample conventional CNN (top).

Appendix

A. CNN Overview

CNNs are simple feedforward ANNs that are simple models of mammalian visual cortex and the same themes have been revealed in the past ten years by the auditory neuroscience that these same design paradigms can be found in the primary and belt auditory areas of the cortex in a number of different animals. Fig. 6 shows a typical CNN structure which is designed for 28 × 28 pixel images. There are subsampling layers between each convolutional layer, which decimate the so-called feature maps of the previous layers neurons. In this sample illustration, the kernel size is set to 5, and the subsampling factors for both dimensions are set to 2. The input layer has the input image (R, G, B) color channels as the feature maps, and after sufficient number of subsampling, the last subsampling layer reaches a scalar (1-D) neurons. Following, CNNs contain fully connected layers that have the same structure as the MLPs.

In order to accomplish this setup, there are several parameters to be set for such typical CNNs: width and height of the input image, kernel (filter) dimensions at each level (usually fixed for each level), the topology of the CNN (number of CNN and MLP hidden layers and number of neurons at each layer), and subsampling factors at each level (usually fixed for each level). Moreover, the input image dimensions should be set according to the number of CNN layers, kernel size, and subsampling factors so that the output CNN layer can reach to a scalar 1 × 1 feature map. One can only train the CNN, once all these parameters are properly fixed in advance. To address these drawbacks, we shall present an adaptive CNN implementation next.

B. Conventional BP

1) BP at MLP layers: In our adaptive CNN implementation, the MLP layers are optional, and if present, the BP operation

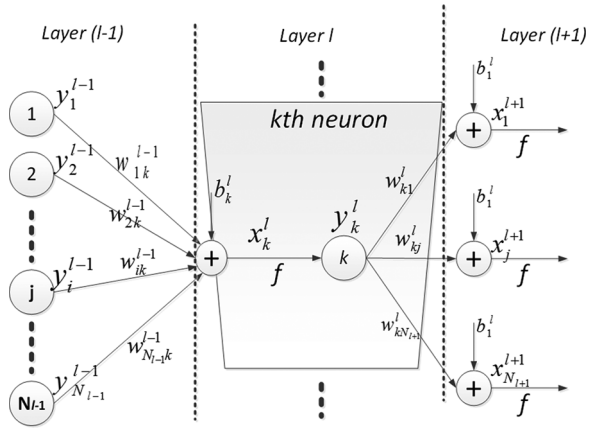

Fig. 7. Three consecutive MLP layers.

starts from the output MLP layer toward the inner layers. Consider a generic MLP illustration in Fig. 7 showing the connections between three layers, $l-1$ to $l+1$ to and from the kth neuron at layer l. The FP of the outputs at layer l–1 toward the output of the kth neuron at layer l can be expressed as follows:

$$x_k^l = b_k^l + \sum_{i=1}^{N_{l-1}} w_{ik}^{l-1} y_i^{l-1} \text{ and } y_k^l = f(x_k^l). \tag{17}$$

Let $l = 1$ and $l = L$ be the input and output layers, respectively. For an input vector p, and its corresponding output vector, $[y_1^L, \ldots, y_{N_L}^L]$, let $[t_1, \ldots, t_{N_L}]$ be the target class vector. The MSE in the output layer can then be expressed as

$$E = E(y_1^L, \ldots, y_{N_L}^L) = \sum_{i=1}^{N_L} \left(y_i^L - t_i\right)^2. \tag{18}$$

We are interested to find out the derivative of this error with respect to an individual weight (connected to that neuron, k) w_{ik}^{l-1}, and bias of the neuron k, b_k^l, so that we can perform gradient descent method to minimize the error accordingly

$$\begin{aligned} \frac{\partial E}{\partial w_{ik}^{l-1}} &= \frac{\partial E}{\partial x_k^l}\frac{\partial x_k^l}{\partial w_{ik}^{l-1}} = \frac{\partial E}{\partial x_k^l} y_i^{l-1} \\ \frac{\partial E}{\partial b_k^l} &= \frac{\partial E}{\partial x_k^l}\frac{\partial x_k^l}{\partial b_k^l} = \frac{\partial E}{\partial x_k^l}. \end{aligned} \tag{19}$$

Both derivatives depend on the sensitivities of the error to the input x_k^l. These sensitivities are usually called as *delta* errors. Let $\Delta_k^l = \frac{\partial E}{\partial x_k^l}$ be the delta error of the kth neuron at layer l. Now, we can write the *delta* error by one step BP from the output of that neuron y_k^l:

$$\Delta_k^l = \frac{\partial E}{\partial x_k^l} = \frac{\partial E}{\partial y_k^l}\frac{\partial y_k^l}{\partial x_k^l} = \frac{\partial E}{\partial y_k^l} f'(x_k^l). \tag{20}$$

This means that the moment we found the sensitivities of the error to the output $\frac{\partial E}{\partial y_k^l}$, we can then find the *delta* error. For the output layer $l = L$, we know both terms

$$\Delta_k^L = \frac{\partial E}{\partial x_k^L} = f'(x_k^L)\left(y_k^L - t_k\right). \tag{21}$$

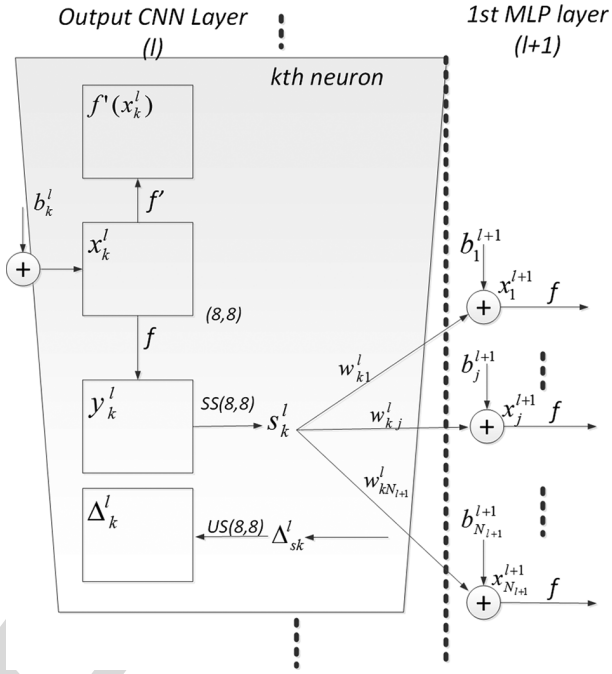

Fig. 8. Output CNN layer connected to the first MLP layer.

Now Fig. 7 considers the output of the previous layer neuron's output y_k^l, which contributes all the neurons' input in the next layer, i.e.,

$$\begin{aligned} x_1^{l+1} &= \ldots + w_{k1}^l y_k^l + \ldots \\ &\ldots \\ x_i^{l+1} &= \ldots + w_{ki}^l y_k^l + \ldots \\ &\ldots \\ x_{N_{l+1}}^{l+1} &= \ldots + w_{kN_{l+1}}^l y_k^l + \ldots . \end{aligned} \tag{22}$$

Therefore, this basically means that the output of the kth neuron in the previous layer y_k^l contributes to the input of the neurons of the current layer with individual weights w_{ki}^l. With this in mind, one can write the sensitivities of the error to the output $\frac{\partial E}{\partial y_k^l}$ as follows:

$$\frac{\partial E}{\partial y_k^l} = \sum_{i=1}^{N_{l+1}} \frac{\partial E}{\partial x_i^{l+1}}\frac{\partial x_i^{l+1}}{\partial y_k^l} = \sum_{i=1}^{N_{l+1}} \Delta_i^{l+1} w_{ki}^l. \tag{23}$$

Interestingly, the same weights are now used to BP the delta errors to the output sensitivity of that neuron, and we already know how to get the delta of that neuron Δ_k^l, from this sensitivity (in (20)), which leads to the generic equation of the BP of the deltas

$$\Delta_k^l = \frac{\partial E}{\partial x_k^l} = \frac{\partial E}{\partial y_k^l} f'(x_k^l) = f'(x_k^l) \sum_{i=1}^{N_{l+1}} \Delta_i^{l+1} w_{ki}^l. \tag{24}$$

This means that the delta of the kth neuron at layer l, Δ_k^l, will be formed by all deltas of the next layer, Δ_i^{l+1}, weighted by the connections from k to i, w_{ki}^l. In other words, if the output of the kth neuron at layer l contributes to a neuron i in the next

level with a weight w_{ki}^l, then next layer neuron's delta Δ_i^{l+1} will contribute with the same weight to form Δ_k^l of the neuron in the previous layer l.

Once all the deltas in each layer are formed by BP, then weights and bias of each neuron can be updated by the gradient descent method. Specifically, the delta of the kth neuron at layer l, Δ_k^l will be used to update the bias of that neuron and all weights of the neurons in the previous layer connected to that neuron as given in (19)

$$\frac{\partial E}{\partial w_{ik}^{l-1}} = \Delta_k^l y_i^{l-1} \quad \text{and} \quad \frac{\partial E}{\partial b_k^l} = \Delta_k^l. \tag{25}$$

In simple words, the sensitivity of the weight connecting the ith neuron in the previous layer to the kth neuron in the current layer depends on the output of the previous layer neuron and the delta of the current layer neuron.

2) BP From the MLP layer to the Output CNN Layer: As illustrated in Fig. 8, the output layer of CNN is connected to the first MLP layer, and, hence, the outputs of this layer CNN neurons are scalars. In other words, s_k^l and of course, Δs_k^l are now all scalars and to achieve this recall that the subsampling factors *ssx* and *ssy*, in this particular layer all are set to the dimensions of the input map. Similarly, the weights of this CNN layer neurons w_{ki}^l are also all scalar and instead of convolution, scalar multiplication is performed as in a regular MLP. So from MLP layer to the CNN layer, the regular BP is simply performed as in (23):

$$\frac{\partial E}{\partial s_k^l} = \Delta s_k^l = \sum_{i=1}^{N_{l+1}} \frac{\partial E}{\partial x_i^{l+1}} \frac{\partial x_i^{l+1}}{\partial s_k^l} = \sum_{i=1}^{N_{l+1}} \Delta_i^{l+1} w_{ki}^l. \tag{26}$$

REFERENCES

[1] R. Hoekema *et al.*, "Geometrical aspects of the interindividual variability of multilead ECG recordings," *IEEE Trans. Biomed. Eng.*, vol. 48, no. 5, pp. 551–559, May 2001.

[2] K. Minami *et al.*, "Real-time discrimination of ventricular tachyarrhythmia with Fourier-transform neural network," *IEEE Trans. Biomed. Eng.*, vol. 46, no. 2, pp. 179–185, Feb. 1999.

[3] Inan *et al.*, "Robust neural-network based classification of PVCs using wavelet transform and timing interval features," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 12, pp. 2507–2515, Dec. 2006.

[4] X. Alfonso and T. Q. Nguyen, "ECG beat detection using filter banks," *IEEE Trans. Biomed. Eng.*, vol. 46, no. 2, pp. 192–202, Feb. 1999.

[5] J. L. Willems and E. Lesaffre, "Comparison of multigroup logisitic and linear discriminant ECG and VCG classification," *J. Electrocardiol.*, vol. 20, pp. 83–92, 1987.

[6] J. L. Talmon, *Pattern Recognition of the ECG*. Berlin, Germany: Akademisch Proefscrift, 1983.

[7] D. A. Coast *et al.*, "An approach to cardiac arrhythmia analysis using hidden Markov models," *IEEE Trans. Biomed. Eng.*, vol. 37, no. 9, pp. 826–836, Sep. 1990.

[8] S. Osowski *et al.*, "Support vector machine based expert system for reliable heartbeat recognition," *IEEE Trans.Biomed. Eng.*, vol. 51, no. 4, pp. 582–589, Apr. 2004.

[9] Y. H. Hu *et al.*, "Applications of artificial neural networks for ECG signal detection and classification," *J. Electrocardiol.*, vol. 26, pp. 66–73, 1994.

[10] Y. Hu *et al.*, "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.

[11] S. C. Lee, "Using a translation-invariant neural network to diagnose heart arrhythmia," in *Proc. IEEE Conf. Neural Inf. Process. Syst.*, Nov. 1989, pp. 2025–2026.

[12] P. de Chazal and R. B. Reilly, "A patient-adapting heartbeat classifier using ECG morphology and heartbeat interval features," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 12, pp. 2535–2543, Dec. 2006.

[13] *Recommended Practice for Testing and Reporting Performance Results of Ventricular Arrhythmia Detection Algorithms*. Arlington, VA, USA: Association for the. Advancement of Medical Instrumentation, 1987.

[14] P. de Chazal *et al.*, "Automatic classification of heartbeats using ECG morphology and heartbeat interval features," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 7, pp. 1196–1206, Jul. 2004.

[15] W. Jiang and S. G. Kong, "Block-based neural networks for personalized ECG signal classification," *IEEE Trans. Neural Networks*, vol. 18, no. 6, pp. 1750–1761, Nov. 2007.

[16] T. Ince *et al.*, "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.

[17] S. Kiranyaz *et al.*, *Multi-Dimensional Particle Swarm Optimization for Machine Learning and Pattern Recognition*. New York, NY, USA: Springer, Aug. 2013.

[18] M. Llamedo and J. P. Martinez, "An automatic patient-adapted ECG heartbeat classifier allowing expert assistance," *IEEE Trans. Biomed. Eng.*, vol. 59, no. 8, pp. 2312–2320, Aug. 2012.

[19] C. Li *et al.*, "Detection of ECG characteristic points using wavelet transforms," *IEEE Trans. Biomed. Eng.*, vol. 42, no. 1, pp. 21–28, Jan. 1995.

[20] T. Mar *et al.*, "Optimization of ECG classification by means of feature selection," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 8, pp. 2168–2177, Aug. 2011.

[21] D. H. Wiesel and T. N. Hubel, "Receptive fields of single neurones in the cat's striate cortex," *J. Physiol.*, vol. 148, pp. 574–591, 1959.

[22] R. Mark and G. Moody. MIT-BIH Arrhythmia Database Directory. [Online]. Available: http://ecg.mit.edu/dbinfo.html

[23] J. Pan and W. J. Tompkins, "A real-time QRS detection algorithm," *IEEE Trans. Biomed. Eng.*, vol. 32, no. 3, pp. 230–236, Mar. 1985.

[24] D. C. Ciresan *et al.*, "Deep big simple neural nets for handwritten digit recognition," *Neural Comput.*, vol. 22, no. 12, pp. 3207–3220, 2010.

[25] D. Scherer *et al.*, "Evaluation of pooling operations in convolutional architectures for object recognition," in *Proc. Int. Conf. Artif. Neural Netw.*, 2010, pp. 92–101.

[26] A. Krizhevsky *et al.*, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. Conf.*, 2012, pp. 1106–1114.

[27] 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.

Serkan Kiranyaz was born in Turkey in 1972. He received the B.S. degree from the Electrical and Electronics Department and the M.S. degree in signal and video processing from Bilkent University, Ankara, Turkey, in 1994, and 1996, respectively, and the Ph.D. degree from the Institute of Signal Processing, Tampere University of Technology, Tampere, Finland, in 2005.

He received the Docent title from the Institute of Signal Processing, Tampere University of Technology, in 2007. He was a Senior Researcher with Nokia Research Center, and later with Nokia Mobile Phones, Tampere. He was a Professor with the Signal Processing Department, Tampere University of Technology, during 2009 to 2015, and he held the Research Director position for the Department and also for the Center for Visual Decision Informatics, Finland. He is currently a Professor at the Department of Electrical Engineering, Qatar University, Doha, Qatar.

He has published two books, more than 35 journal papers on many IEEE Transactions and other high impact journals and around 80 papers in international conferences. His principal research interests include "intelligent systems" that comprises systems, devices, algorithms, software and processes requiring intelligent processing of, possibly "big" data, or sensor signals together with artificial reasoning. He made significant contributions on biosignal analysis, particularly EEG and ECG analysis and processing, classification and segmentation, signal processing and management, computer vision, scene analysis, automatic object segmentation, and machine learning strategies with applications to recognition, classification, multimedia retrieval, evolving systems and evolutionary machine learning, swarm intelligence, stochastic optimization, and deep learning.

Turker Ince received the B.S. degree from the Bilkent University, Ankara, Turkey, in 1994, the M.S. degree from the Middle East Technical University, Ankara, in 1996, and the Ph.D. degree from the University of Massachusetts Amherst (UMass Amherst), Amherst, MA, USA, in 2001, all in electrical engineering.

From 1996 to 2001, he was a Research Assistant with the Microwave Remote Sensing Laboratory, UMass Amherst. He was a Design Engineer with Aware, Inc., Boston, USA, from 2001 to 2004, and with Texas Instruments, Inc., Dallas, USA, from 2004 to 2006. In 2006, he joined the Faculty of Engineering and Computer Science, Izmir University of Economics, Izmir, Turkey, where he is currently an Associate Professor. His research interests include electromagnetic remote sensing and target recognition, radar signal processing, biomedical signal processing, artificial neural networks, and evolutionary optimization techniques.

Moncef Gabbouj (F'xx) received the B.S. degree in electrical engineering from Oklahoma State University, Stillwater, OK, USA, in 1985, and the M.S. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1986 and 1989, respectively. Q2

He held several visiting professorships with different universities, including The Hong Kong University of Science and Technology, Hong Kong; Purdue University; the University of Southern California; and the American University of Sharjah, UAE. He was the Head of the Department during 2002–2007, and was the Senior Research Fellow with the Academy of Finland in 1997–1998 and 2007–2008. He is currently the Academy of Finland Professor of signal processing at the Department of Signal Processing, Tampere University of Technology, Tampere, Finland. He coauthored more than 600 publications and supervised 40 Ph.D. dissertations. He is internationally recognized for his research in the areas of nonlinear signal and image processing and analysis. In addition, his research interests include multimedia analysis, indexing and retrieval, machine learning, voice conversion, and video processing and coding.

Dr. Gabbouj is a Member of the Academia European and the Finnish Academy of Science and Letters. He is a Member of the IEEE Fourier Award, and the Past-Chairman of the DSP Technical Committee of the IEEE Circuits and Systems Society. He was an Honorary Guest Professor of Jilin University, China, during 2005–2010. He served as a Distinguished Lecturer for the IEEE Circuits and Systems Society in 2004–2005, and the Past-Chairman of the IEEE-EURASIP Nonlinear Signal and Image Processing Board. He was the Chairman of the Algorithm Group of the EC COST 211quat. He served as an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, and was the Guest Editor of the *Multimedia Tools and Applications*, the *European Journal Applied Signal Processing*. He is the Past-Chairman of the IEEE Finland Section and the IEEE SP/CAS Finland Chapter. He was also the Chairman and Technical Program Chair of many national and international conferences and workshops. He is a member of IEEE SP and CAS Societies. He received the 2012 Nokia Foundation Visiting Professor Award, the 2005 Nokia Foundation Recognition Award, and many best paper awards.

QUERIES

Q1. Author: Please provide the year in Ref. [22].

Q2. Author: Please provide the year in which the author "Moncef Gabbouj" became a Fellow of the IEEE.