Support Vector Machine and Artificial Neural Network Implementation in Embedded Systems for Real Time Arrhythmias Detection

Andrés Orozco-Duque[1], Santiago Rúa[1], Santiago Zuluaga[1], Alfredo Redondo[1], Jose V. Restrepo[2] and John Bustamante[1]

[1]*Centro de Bioingenieria, Universidad Pontificia Bolivariana, Cr. 1 70-1, Medellín, Colombia*

[2]*Grupo de Investigación en Microelectrónica, Universidad Pontificia Bolivariana, Cr. 1 70-1, Medellín, Colombia*

Keywords: Arrhythmias, Artificial Neural Network, ECG Signal, FPGA, Microcontroller, Support Vector Machine.

Abstract: This article presents the development and implementation of an artificial neural network (ANN) and a support vector machine (SVM) on a 32-bit ARM® Cortex® M4 microcontroller core from Freescale Semiconductor and on a FPGA Spartan® 6 from Xilinx™, looking for real-time detection of ventricular tachycardia (VT) and ventricular fibrillation (VF). They were compared in terms of accuracy and computational cost. A Fast Wavelet Transform (FWT) was used, and the energy in each sub-band frequency was calculated in the feature extraction stage. For the training and validation algorithms, labeled signals from MIT-BIH database with normal sinus rhythm, VF and VT in a time window of 2 seconds were used. Test results achieve an accuracy of 99.46% for both ANN and SVM with execution time less than 0.6 ms in microcontroller and 30 μs in FPGA for ANN and less than 30 ms in a microcontroller for SVM. The test was done with a 32 MHz clock.

1 INTRODUCTION

Wearable Cardiac devices are currently used to monitor the ECG signal in patient with high cardiac risk and detect arrhythmias in real-time (Oresko et al., 2010). Pre-processing, feature extraction and classification are the main processing stages in this kind of devices.

Ventricular Fibrillation (VF) is considered a fatal arrhythmia and Ventricular Tachycardia (VT) may induce VF over time. For this reason, a real-time cardiac monitoring systems must to include a module for the detection of these arrhythmias. To detect VT and VF in real-time, various algorithms have been developed, such as fuzzy similarity-base approximate entropy (Xie et al., 2011), Wavelet based features (Balasundaram et al., 2011), phase space reconstruction (Sáenz and Bustamante, 2009), feature selection with Support Vector Machine (SVM) (Alonso-Atienza et al., 2012) and Artificial Neural Network (ANN) (Valenza et al., 2008).

For implementing of these algorithms in embedded systems such as FPGA or microcontroller, an analysis of the computational cost is necessary. This paper compares the accuracy of two machine learning methods: ANN and SVM. In order to measure and compare the computational cost, both algorithms were implemented in a 32-bit microcontroller, further an ANN was implemented in the FPGA.

2 METHODOLOGY

2.1 Fast Wavelet Transform

For the pre-processing and feature extraction stage, the Fast Wavelet Transform (FWT) was used. FWT works like an array of high pass $h(n)$ and low pass $l(n)$ filters where coefficients depend of the mother wavelet selected and with cut frequency set in the middle of the input signal's spectrum (Benedetto, 1999). The block diagram of FWT is shown in the Figure 1.

FWT was developed on the FPGA. In this implementation, is important to synchronize the coefficients of the different levels because the delay increases as the level of decomposition.

Symlets 8 was selected as mother wavelet according to (Singh and Tiwari, 2006).

Orozco-Duque A., Rúa S., Zuluaga S., Redondo A., V. Restrepo J. and Bustamante J..
Support Vector Machine and Artificial Neural Network Implementation in Embedded Systems for Real Time Arrhythmias Detection.
DOI: 10.5220/0004232003100313
In *Proceedings of the International Conference on Bio-inspired Systems and Signal Processing* (BIOSIGNALS-2013), pages 310-313
ISBN: 978-989-8565-36-5

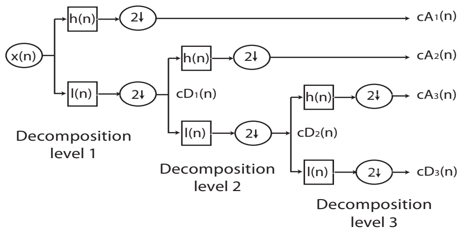

Figure 1: Block diagram of FWT.

2.2 Sub-bands Wavelet Energy

The ECG signal was segmented in two seconds time window. The energy contribution of the Wavelet's sub-bands coefficients were extracted to create the feature vector. Wavelet coefficients of decomposition level cD3, cD4 and cD5, and approximation level A5 were selected. Other coefficients were discarded because they represent less than 2% of the total signal's energy.

2.3 Signal Database

MIT-BIH Malignant Ventricular Arrhythmia Database and MIT-BIH Normal Sinus Rhythm Database (Goldberger et al., 2000) were used. Seven different labeled signals of normal ECG, five signals with VT and 4 with VF were downloaded. 1163 segments with VT were extracted, 1292 with VF and 813 with normal rhythm. 60 % of the segments were located in the training group, 20 % in the cross validation group and the last 20 % in the testing group.

2.4 Hardware

A Kinetis Microcontroller from Freescale Semiconductors™ was selected for its low-power performance with an ARM® Cortex M4 core with Floating Point (FP) unit and DSP functions.

The selected FPGA was a Spartan® 6 XC6SLX9 from Xilinx™ for its balance of low cost, and low power consumption. It has 9152 logic cells, 1430 slices, 11440 flip-flops, 32 of 18kb Block RAM and 16 DSP slices.

3 ALGORITHMS IMPLEMENTATION

3.1 Artificial Neural Network

A feed-forward ANN with four layers was implemented: one input layer with four neurons, two hidden layers each with three neurons, and one output layer with only one neuron. This structure was selected after different test with a cross validation set using MATLAB®. Each neuron of the input layer corresponds to one of the energy sub-bands obtained from FWT. The output layer classifies between normal rhythm and VT/VF (Figure 2).

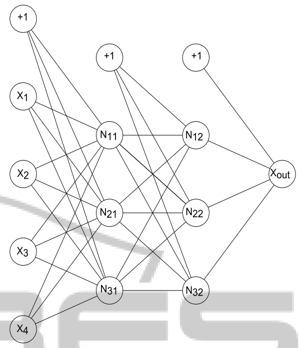

Figure 2: ANN structure designed for VT/VF detection.

The ANN was trained in MATLAB using the back-propagation algorithm of the Neural Network Toolbox. The model obtained including the bias matrix is:

$$W_{layer1} = \begin{bmatrix} 13.6942 & 0.4326 & 0.0403 & 0.0823 & 0.7342 \\ 2.5106 & -4.631 & 0.2119 & 0.6442 & 0.4936 \\ 9.6895 & -0.9629 & -0.6307 & 2.0219 & -0.2652 \end{bmatrix} \tag{1}$$

The weights matrix for layer two is:

$$W_{layer2} = \begin{bmatrix} 17.2050 & -1.4175 & 2.4593 & 6.1326 \\ 5.3072 & -0.8866 & -8.6401 & 4.4102 \\ -1.8915 & 6.6518 & 4.4547 & 4.4037 \end{bmatrix} \tag{2}$$

The output layer is:

$$W_{layout} = \begin{bmatrix} 3.0015 & -3.8829 & -2.2100 & 3.0934 \end{bmatrix} \tag{3}$$

In the matrix W_{layer1}, W_{layer2} and W_{layout}, the columns represent the weights of each input of a neuron, each row represent one neuron in the layer.

The *pureline* activation function was used in the output layer and *sigmoid* function was used in the hidden layers.

The ANN was implemented in the FPGA using System Generator™ from Xilinx™. This software allows the use of the Simulink™ development environment to designing high-performance DSP systems. FPGA allows parallel processing by reconfiguring the hardware according to the program. Figure 3 shows the architecture of one neuron implemented in System Generator™.

Sigmoid activation function is given by

$$P(t) = \frac{1}{1+e^{-t}} \tag{4}$$

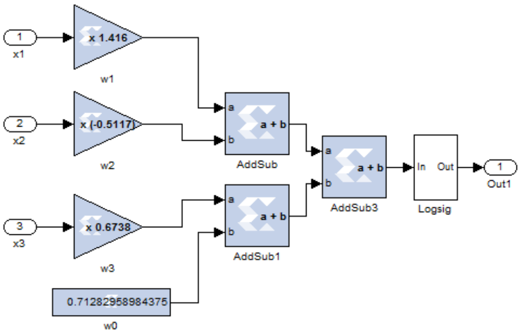

Figure 3: Artificial neuron implemented in System Generator.

To implement this function in the FPGA a step-by step approximate function was used. The sigmoid approximate activation function used was:

$$a = \begin{cases} 0, & x < 0 \\ 0.1572, & -2 \leq x < -1 \\ 0.334, & -1 \leq x < 0 \\ 0.5, & x = 0 \\ 0.5734, & 0 < x \leq 0.8 \\ 0.7356, & 0.8 < x < 1.5 \\ 0.8779, & 1.5 \leq x \leq 2.5 \\ 0.9509, & 2.5 < x \leq 3.5 \\ 1, & 3.5 < x \end{cases} \quad (5)$$

This was implemented using a ROM block, which stored the outputs of the activation function. The system was designed to select the address of the ROM memory according with the input of the activation function.

The ANN model was implemented using System Generation with the artificial neuron as shown in Figure 3.

In the microcontroller, the algorithm was implemented on *C*. The CMSIS 2.0 library was used in the ANN implementation, which has more than 60 DSP functions including fixed point and single-precision floating point math. Using this library is possible to implement the prediction algorithm of ANN with matrix operations as shown in the following code:

```
arm_mat_mult_f32(&L1,&C,&NL1);
arm_mat_add_f32(&NL1,&B1,&NL1);
activation_sigmoid(&ANN_layer1[0],NN_layer1);
```

Where `&C`, `&L1`, `&B1`, `&NL1` are pointers to feature matrix, weights matrix of layer 1, bias vector for layer 1 and ANN structure for neurons of layer 1 respectively. *Arm_mat_mult* and *arm_mat_add* are matrix operations of CMSIS 2.0 and *activation_sigmoid* is a function developed by the authors. This piece of code calculates the propagation through layer 1 of the ANN. To implement the complete feed-forward propagation algorithm, the code is the same but with iterations according to the number of layers.

The sigmoid activation function was implemented *C* using "math.h" library, it was not necessary to use the approximation as with the FPGA.

3.2 Support Vector Machine (SVM)

A SVM prediction algorithm was implemented in the microcontroller. SVM is a supervised learning machine method which selects the hyperplane that divides the training set into two classes so that maximizes distance of each set (Steinwart and Christmann, 2008). In a SVM, given X, an input vector with m dimensions, a new feature vector $f \in R^{m-1}$ is computed depending of the proximity to vectors in the training set. To compute the proximity, a Gaussian kernel expression:

$$k(x, x') = e^{\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right)} \quad (6)$$

was used. The prediction is calculated with the inner product between the new feature vector f and the weights (alphas) of the model θ^T, where the prediction is 1, if $\theta^T f \geq 0$.

The SVM training was executed in MATLAB with the training set referenced in section 2.3 using a simplified version of SMO algorithm from the Learning Machine online course of Andrew Ng from Stanford University. The regularization parameter and the σ parameter of the Gaussian kernel were optimized using the cross validation set to avoid high bias or high variance.

The SVM prediction function was implemented in a microcontroller on *C* language using CMSIS 2.0 library and similar methodology as on the ANN implementation.

The SVM implemented in this work is a four dimensions machine.

4 RESULTS AND DISCUTION

Accuracy of the detection algorithm in the test set was 99.46% for both, ANN and SVM, measure was done after implementation on FPGA and Microcontroller.

ANN was implemented in both, 32 bit microcontroller and FPGA, table 1 shows the execution times.

Table 1: ANN prediction time.

Embedded System	**Time** (μ *s*)
Microcontroller (32Mhz)	600
FPGA (32Mhz)	0.30

SVM was implemented only in the microcontroller. Table 2 shows the execution times of SVM and ANN in the microcontroller.

Table 2: ANN and SVM prediction time in microcontroller.

Algorithm	Time (ms)
ANN	0.6
SVM	30

SVM and ANN can reach same accuracy; however prediction algorithm for SVM takes about 50 times more than ANN in the same microcontroller. This is shown in Table 2. For this application ANN is better because the execution time is less than SVM with the same accuracy.

In this application, there is a time limit of two seconds determined by the time windows used in the feature extraction stage. Table 2 show that VF/VT could be detected in real time using both SVM and ANN.

The support vector machine was only implemented in the microcontroller. The implementation of SVM in an FPGA requires the realization of complex mathematical operations for calculating the Gaussian kernel. Its implementation in the FPGA using System Generation is difficult because the multiplications consumes all the resources of DSP-slices which are limited. It is necessary the used of a math processor tha could be implemented in a FPGA.

The microcontroller's advantages are the low power consumption and the additional peripherals. Furthermore, it allows easy programming of complex mathematical operations because is programmed in C.

Currently, the authors are working on multiclass classifier that will allow to differentiate between VT and VF and to detect other types of arrhythmias, such as premature atrial contractions, premature ventricular contractions, atrial fibrillation, atrial flutter, among others. Therefore the develop of robust classifier in real time is necessary. A mixed platform with a microcontroller and parallel co-processors implemented on a FPGA has been developed to implement a real-time pre-processing, feature extraction and classification system.

5 CONCLUSIONS

In this work, two machine learning methods were implemented in an embedded system: SVM and ANN. Both methods can be used to detect VT/VF in real-time with FWT scales energy as features and 2 seconds windows analysis.

SVM and ANN are powerful tools for arrhythmias classification and could be real-time implemented in both, microcontroller and FPGA. ANN is faster than SVM, for VT/VF detection is better use ANN because it has lower execution time than SVM, nevertheless SVM was implemented considering future work with other types of arrhythmias, where a robust classifier will be necessary.

REFERENCES

Alonso-Atienza, F., Rojo-Álvarez, J. L., noz, A. R.-M., Vinagre, J. J., García-Alberola, A., and Camps-Valls, G. (2012). Feature selection using support vector machines and bootstrap methods for ventricular fibrillation detection. *Expert Systems with Applications*, 39(2):1956 – 1967.

Balasundaram, K., Masse, S., Nair, K., Farid, T., Nanthakumar, K., and Umapathy, K. (2011). Wavelet-based features for characterizing ventricular arrhythmias in optimizing treatment options. In *Engineering in Medicine and Biology Society,EMBC, 2011 Annual International Conference of the IEEE*, pages 969 –972.

Benedetto, J. J. (1999). Sampling theory and wavelets. In *Signal Processing for Multimedia*, pages 19–33.

Goldberger, A. L., Amaral, L. A. N., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., and Stanley, H. E. (2000). Physiobank, physiotoolkit, and physionet. *Circulation*, 101(23):e215–e220.

Oresko, J., Jin, Z., Cheng, J., Huang, S., Sun, Y., Duschl, H., and Cheng, A. (2010). A wearable smartphone-based platform for real-time cardiovascular disease detection via electrocardiogram processing. *Information Technology in Biomedicine, IEEE Transactions on*, 14(3):734 –740.

Sáenz, J. F. and Bustamante, J. (2009). Algoritmo computacional para la detección de fibrilación ventricular basado en la técnica de reconstrucción de espacio de fases con tiempo de retardo variable. *Revista de la Federación Argentina de Cardiología*, 38:60–65.

Singh, B. N. and Tiwari, A. K. (2006). Optimal selection of wavelet basis function applied to ecg signal denoising. *Digital Signal Processing*, 16(3):275 – 287.

Steinwart, I. and Christmann, A. (2008). Support vector machines. Springer.

Valenza, G., Lanata, A., Ferro, M., and Scilingo, E. (2008). Real-time discrimination of multiple cardiac arrhythmias for wearable systems based on neural networks. In *Computers in Cardiology, 2008*, pages 1053 –1056.

Xie, H.-B., Zhong-Mei, G., and Liu, H. (2011). Classification of ventricular tachycardia and fibrillation using fuzzy similarity-based approximate entropy. *Expert Systems with Applications*, 38(4):3973 – 3981.