SVM SPEAKER VERIFICATION SYSTEM BASED ON A LOW-COST FPGA

Rafael Ramos-Lara, Mariano López-García, Enrique Cantó-Navarro

Technical University of Catalonia
08800 Vilanova i Geltrú Spain
email: {lara, lopezg}@eel.upc.edu, canto@etse.urv.es

Luís Puente-Rodriguez

Universidad Carlos III de Madrid
28911, Leganes, Spain
email: lpuente@it.uc3m.es

ABSTRACT

Biometric systems, characterized by their high confidential levels of security, are usually based on high-performance microprocessors implemented on personal computers. These advanced devices contain floating-point units able to carry out millions of operations per second at frequencies in the GHz range, being qualified to resolve the most complex algorithms in just a few hundred of milliseconds. However, their main drawback is the cost, and the necessary space required to incorporate their external associated peripherals. This disadvantage is especially significant in the low-cost consumer market, where factors such as price and size determine the viability of a product. The use of an FPGA is a suited way to implement systems that require a high computational capability at affordable prices. Besides, these devices allow the design of complex digital systems with outstanding performances in terms of execution times. This paper presents the implementation of a SVM (Support Vector Machines) speaker verification system on a low-cost FPGA. Experimental results show as our system is able to verify a person's identity as fast as a high-performance microprocessor based on a Pentium IV personal computer.

1. INTRODUCTION

Recent advances in the field of microelectronics have contributed to developing powerful microprocessors able to reduce the execution time of programs with a high computational cost. These advances have allowed increasing the complexity of algorithms that might use a high number of transcendental functions or floating-point operations, making possible to work with high volumes of information without affecting the execution times.

The development of biometric algorithms has mainly been focused on improving the performances in terms of FRR (False Rejection Rate), FAR (False Acceptance Rate) or ERR (Equal Error Rate). Generally, designers concentrate their design effort in achieving robust and reliable systems, assuming that the hardware implementation is carried out in a microprocessor with enough computational capacity to process all the information in real time, relegating to a second plane other aspects such as cost or space.

Products related to the low-cost consumer market are normally affected by hardware limitations. Usually their design is based on an embedded standard medium-performance microprocessor which sequentially executes a set of operations that are part of a specific algorithm. This simple software implementation is quite flexible, but however not always offers the best performances, particularly in applications featured (as some biometric algorithms) by a high-computational load [1], [2].

Speaker verification is a very well-known biometric modality in which the samples of voice are acquired by using a low-cost sensor device. On the other hand, techniques based on Support Vector Machines (SVM) have shown good results with acceptable recognition rates [3]. There are some publications dealing with FPGA (Field Programmable Gate Array) implementations of generic SVM for classification purpose, but only a reduced group of them are specifically devoted to speaker verification. In [4], authors show the main features of an implementation of a SVM speaker verification system for Match-on-Card. Due to the smart-card memory limitations, authors used the time average of all speech frames as feature vector, representing each utterance by a single 24-dimensional vector. Additionally, the paper shows a FPGA implementation of the matching stage using a kernel based on an exponential function. The system is able to carry out the matching between the model and the feature vector 50 times faster than a software-based solution running on a Pentium IV clocked at 1.3 GHz. Other publications show only hardware implementations of some specific part of algorithms for speech recognition or speaker identification, that allow a significant acceleration of the processing time [5], [6].

This paper presents the implementation of a whole SVM speaker verification system based on dedicated hardware. The system consists of several stages dedicated to calculate the feature vectors, based on Mel-frequency Cepstral coefficients and their associated deltas, as well as the matching between these vectors and the user's model stored in an external SRAM memory. Experimental results show the viability and the main performances of the proposed hardware implementation made on a low-cost Spartan 3 FPGA.

978-1-4244-3892-1/09/$25.00 ©2009 IEEE

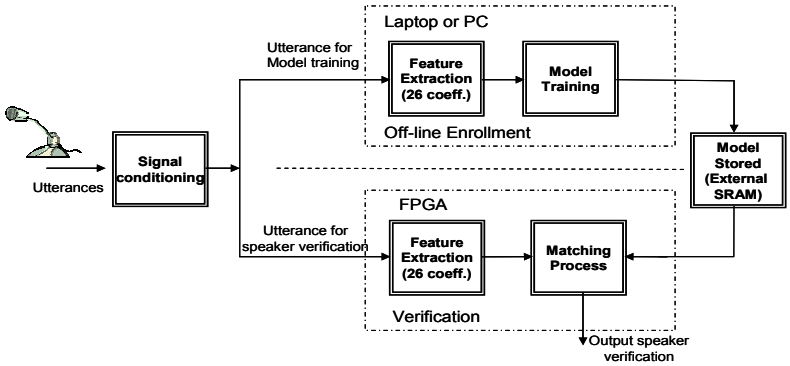

Fig. 1. Block diagram for feature extraction and matching.

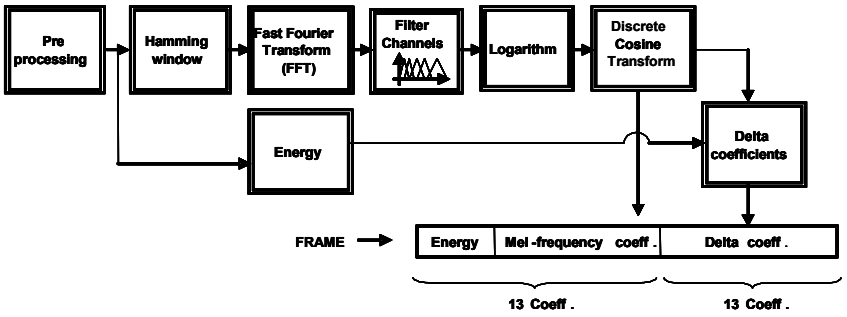

Fig. 2. Block diagram used to obtain the Mel-frequency coefficients.

The paper is organized as follows. Section II reviews briefly the basic theory about the feature extraction process based on Mel-frequency coefficients and the SVM classifier. Section III presents the internal architecture of the whole system, remarking the main characteristics of the implementation and finally section IV shows the experimental results.

2. FEATURE EXTRACTION

Fig. 1 presents the block diagram of the proposed system. The parameters of the model used by the SVM are obtained by means of a training process based on isolated sentences of approximately 14 seconds. The data used in this process consist of four repetitions of each utterance from 52 speakers (26 males and 26 females). The sample frequency for each speech was 8 kHz with a codification rate of 12 bits.

The size and cost of a FPGA depend directly on the area occupied by the implemented hardware, so that it is important to reduce the number of blocks embedded in the device. The training process is carried out only once, obtaining a model valid for a specific speaker. Afterwards, this model is stored in a memory and used during the matching stage when a claimed identity is presented to be verified. Thus, in order to minimize the FPGA resources needed to build the hardware, the training process is made on a desk PC that executes the algorithm that allows the calculation of those parameters that characterize the user's voice.

During the feature extraction stage, the speech signal is segmented into frames of 25 ms (200 samples), using a frame advance of 10 ms (80 samples) (overlapping of 15 ms). Each frame has been represented by 12 Mel-frequency Cepstral coefficients (see Fig. 2), as well as the energy and their associated deltas (leading to a total of 26 coefficients). Fig. 2 shows the block diagram used to calculate the Mel-frequency coefficients.

The verification process was executed on two different platforms as an example of high and medium performance microprocessors: an Intel Pentium IV at 1.5 GHz and Microblaze at 40 MHz, respectively. Table 1 shows the execution times for those blocks presented in Fig. 2, along with the calculation of their delta coefficients and the matching between the model and a generic user (these results are presented by frame, so that the whole execution time can be straightforward obtained considering the total number of frames analyzed in each utterance). The forth column of this table shows the execution time when the whole system is implemented on dedicated hardware proposed in this paper. Results shown using Microblaze

Table 1. Execution speeds for feature extraction and matching stages by frame on two different microprocessors and the dedicated FPGA hardware.

Function	Execution time on Intel Pentium IV at 1.5 GHz	Execution time on Microblaze at 40 MHz	Execution time on dedicated FPGA hardware at 50 MHz
Pre-processing	14.12 µs	3.13 ms	31.96 µs
Hamming window	3.13 µs	151 µs	24 µs
Fast-Fourier Transf.	63.36 µs	8.83 ms	30.22 µs
Filter Channels	45.45 µs	6.75 ms	116.48 µs
Logarithm	8.41 µs	17.30 ms	53.78 µs
DCT	102.57 µs	216.32 ms	26.46 µs
Delta coefficients	1.73 µs	620 µs	2.54 µs
Frame execution time for feature extraction	*238.77 µs*	*253.1 ms*	*285.44 µs*
Frame matching	*4370.15 µs*	*2304 ms*	*4362 µs*
Total frame execution time	**4608.92 µs**	**2557.1 ms**	**4647.44 µs**

Table 2. Selected values for M and N to carry out operations in fixed-point format for feature extraction and matching.

Function		M (bits integer part)	N (bits fractional part)
Pre-processing		15	8
Hamming window		15	9
Fast Fourier T.	DFT. Coeff.	23	9
	Power of 2 (Re+Im)	42	8
	Module DFT	21	10
Filter Channels		21	2
Logarithm		6	14
Inverse DCT		6	18
Delta coefficients		6	14
Matching		18	31

have been obtained implementing it on a Spartan 3 FPGA. As it can be seen, the Intel Pentium IV takes about 4.6 ms to process a frame, whereas Microblaze makes the same processing in 2557 ms.

Since the system shown in Fig. 1 initiates a new frame each 10 ms, only the high-performance microprocessor is able to carry out the feature extraction and matching frame in real-time. A drawback of executing this processing on Microblaze (and probably in any embedded medium-performance microprocessor), is that it would be necessary to store the utterance in a memory. Afterwards the microprocessor has to begin to read and to process frames according to its computational capability, which leads to an additional increasing of the execution time (storing plus processing).

3. SYSTEM ARCHITECTURE AND HARDWARE IMPLEMENTATION

The source code employed to perform the feature extraction and matching of each frame is written using floating-point computations, due to the presence of functions such as root square, logarithm, exponential and trigonometric. In terms of size and latency, the design of hardware coprocessors able to make floating-point operations is usually complicated. Thus, generally the utilization of floating-point operations in dedicated hardware is only justifiable in certain cases where the result requires a high accuracy or values have wide dynamic ranges.

All the operations involved in our hardware design have been done in fixed-point format, where each real variable consists of an integer part and a fractional part, each one represented by a number of M and N bits, respectively. The error generated in a specific operation is defined as the discrepancy between the exact floating-point value and the approximation obtained in fixed-point format, which depends on the values chosen for M and N. These values are different for each of those blocks shown in Fig. 2, as well as in the calculation of the exponential evaluated in the matching stage. Table 2 shows the selected values that, as experimental results will show, allow the error tends to zero

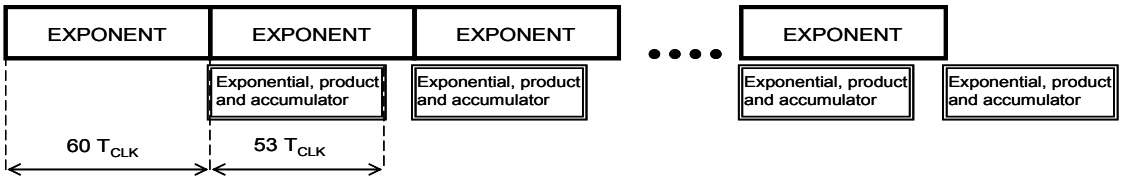

Fig. 3. Scheduling in the calculation of different tasks in expression (1).

Table 3. Device utilization summary for Spartan 3 (the number of slices for matching stage include the synthesis of Picoblaze which is necessary only for testing purpose).

Function	Number of Slices	Number of Flip-Flops	Number of 4 input LUT	Number of Multipliers
Feature extraction	*3817*	*4659*	*6138*	*13*
Matching	*575*	*692*	*647*	*8*

(when an operation uses more than one computation, only the most restrictive values for M and N are presented).

The trigonometric function used in the resolution of DCT has been tabulated in a look-up table of 312 elements codified with 16 bits. On the other hand, the logarithm, root and exponential functions have been implemented using the algorithm described in [7].

As Table 1 shows, due to the radial basis kernel and the number of support vectors used in the matching process, this is the most critical task in terms of execution time. Basically, the comparison between the model and the feature vector is calculated evaluating the following expression:

$$G = \sum_{j=1}^{K} p_j \cdot e^{-\gamma \sum_{i=1}^{26} (x_i - z_{ji})^2}, \qquad (1)$$

where K is the number of z_{ji} support vectors (in our case 3634), p_j are the Lagrange's coefficients obtained in the training process, γ is a constant and x_i is the feature vector that consists of 26 elements.

The design of this dedicated hardware is oriented in order to minimize its execution time following the pipeline scheme shown in Fig. 3. Using this planning regarding different operations, the time needed to achieve a proper classification can be approximated by the time needed to evaluate the exponent of function (1). The calculation of this exponent is done in 60 clock cycles, whereas the exponential, the multiplication by coefficient p_j and the sum and accumulation is carried out in just 53 clock cycles. In order to achieve this target, during the evaluation of these four operations related to support vector j, the assessment of the following exponent related to support vector j+1 is started, leading to a total execution time of:

$$\text{Execution time /frame} = K * 60\, T_{CLK} + 53\, T_{CLK}, \qquad (2)$$

4. EXPERIMENTAL RESULTS

The design of the system has been described in VHDL language and implemented on a Xilinx FPGA Spartan 3 XCS2000. The results of the synthesis in terms of area for feature extraction and matching are presented in Table 3 (about the 24% of the total size of the FPGA). The selected frequency was 50 MHz. Note as almost all the FPGA resources are consumed by the feature extraction hardware which occupies about the 88% of the CLB slices and the 60% of the internal multipliers.

In order to probe the proper operation of the hardware implementation the original algorithm was re-programmed, generating a new version in fixed-point format using the values for M and N indicated in Table 2.

Only for testing purpose an 8 bit microprocessor (Picoblaze) is implemented in the FPGA, which allows the communication via RS-232 between the dedicated hardware and an external laptop (see Fig. 4).

After analyzing several utterances of approximately 1394 frames each one the average relative error in the calculation of the exponential function between the float and fixed-point versions was about 0.51% with a variance of 9%. Fig. 5 shows the histogram obtained upon the comparison of both results. Only the 3.3% of the feature vectors processed gave an error higher than 5%. The exact value of these vectors is closed to zero, which theoretically involves a relative error tends to infinite. Moreover, it is important to remark that for all the utterances tested, none of the errors produced in the calculation of the exponential function lead to an error in the classification process (the

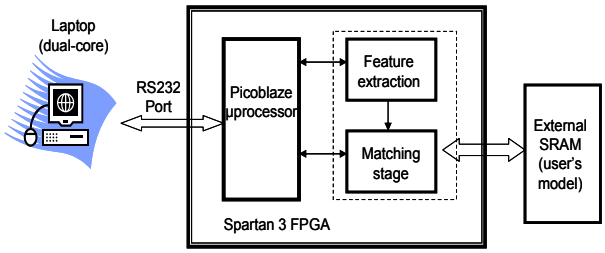

Fig. 4. Block diagram to implement the communication between the FPGA and the laptop.

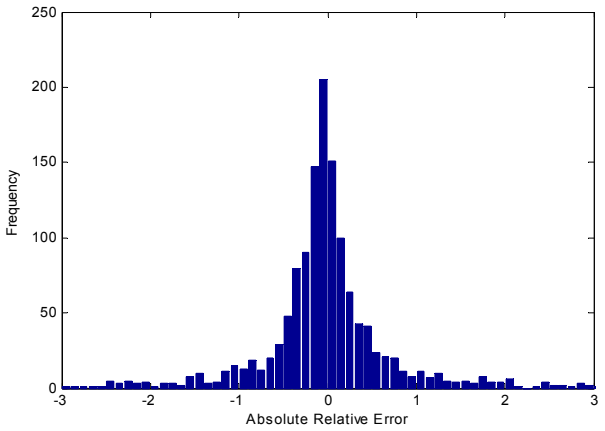

Fig. 5. Histogram representing the relative error in the calculation of the exponential function.

error was produced in the magnitude of the value but not in the sign), due to the proper selection of values in M and N.

On the other hand, the execution time per frame obtained in dedicated hardware is lower than 10ms (the advance frame time). The feature vector is processed in 285.44 µs and the matching between this vector and the model stored in an external SRAM memory is carried out in 4362 µs (each frame is processed in 4647.44 µs). These results are very similar to those obtained with the Intel Pentium IV microprocessor presented in Table 1 (238 µs and 4370 µs).

5. CONCLUSIONS

The implementation of biometric systems is generally done using high-performance microprocessors able to solve complex algorithms in several microseconds. However, when low-cost microprocessors are used, these algorithms may not be solved in real-time due to their limitation for processing functions with a high computational cost. This paper presented the implementation in a FPGA of a speaker recognition system based on SVM. The system was implemented on a low-cost Spartan 3 FPGA clocked at 50MHz, obtaining similar performances, in terms of execution time, to those achieved with a Pentium IV PC. The proposed system is able to carry out a feature vector extraction and its matching against a data base of 3634 vectors in 285 µs and 4362 µs, respectively. The system's design was done implementing all the operations in fixed-point format, in order to reduce the area needed in the FPGA, and selecting the suitable number of bits for the fractional and integer part to minimize the relative error.

6. ACKNOWLEDGEMENTS

Authors thank financial support from Ministerio de Educación y Ciencia de España, under grant TEC2006-12365-C02-02.

7. REFERENCES

[1] Lopez, M., Cantó, E., "FPGA implementation of a Minutiae Extraction Fingerprint Algorithm," *IEEE International Symposium on Industrial Electronics*, Cambridge, U.K. (2008).

[2] Cantó, E., Canyellas, N., Fons, M., Fons, F., López, M., "FPGA Implementation of the Ridge Line Following Fingerprint Algorithm," *14th International Conference on Field-Programmable Logic and Applications*, Springer-Verlag LNCS 3203, Antwerp, Belgium (2004), pp. 1087-1089.

[3] Burges, C.J.C., "A Tutorial on Support Vector Machines for Pattern Recognition," *1998 Kluwer Academic Publishers, Data Mining and Knowledge Discovery*, vol. 2, pp. 121-167.

[4] Choi, W-Y., Ahn, D., Burn Pan, S., Chung, K., Chung, Y., Chung, S-H. "SVM-Based Speaker Verification System for Match-on-Card and its Hardware Implementation", *ETRI Journal*, vol. 28, no. 3, pp. 320-328, June 2006.

[5] Nedevschi, S., Patra, R., Brewer, E., "Hardware Speech Recognition for User Interfaces in Low cost, Low Power Devices," *43nd Design Automation Conference*, IEEE Press, California, June 2005, pp.684-689.

[6] Melnikoff, S., Quigley, S.F., Rusell, M. J., "Implementing a Simple Continuous Speech Recogniton System on an FPGA," *Proceedings of the 10th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, Napa, California, USA (2002).

[7] Ercegovac, M.D., "Digital Aritmetic," Ed. Morgan Kaufmann (2004).