Parallel implementation for wavelet dictionary optimization applied to pattern recognition*

Leandro D. Vignolo[1], Diego H. Milone[1,2],
Hugo L. Rufiner[1,2] y Enrique M. Albornoz[1]

[1] Grupo de Investigación en Señales e Inteligencia Computacional.
Facultad de Ing. y Cs. Hídricas, Universidad Nacional del Litoral, Argentina
[2] Laboratorio de Cibernética,
Facultad de Ingeniería, Universidad Nacional de Entre Ríos, Argentina
{ldvignolo, dmilone, lrufiner, emalbornoz}@fich.unl.edu.ar

Abstract. By means of full wavelet packet decomposition a redundant set of coefficients is obtained. For signal classification it is convenient to find a subset of these coefficients minimizing the error rate of a classifier. A problem arises because of the computational cost of GA solution. This work presents the parallelization of a genetic algorithm by which it is possible to obtain the best subset of coefficients in order to improve results on phoneme recognition. Various strategies have been evaluated in order to improve the classifier initialization and the evolution itself. Classification results for a set of Spanish phonemes show the advantages of the propossed method and the speedup of the implemented parallelization.

1 Introduction

Automatic speech recognition systems need a preprocessing stage to make the key features of the phonemes more evident, allowing to improve classification results [16]. This task is usually accomplished by different signal processing techniques like filter banks, linear prediction and cepstrum analysis [17].

Wavelet transform characteristics make this tool usefull for non stationary signal analysis [11]. Multiresolution analysis asociated with discrete wavelet transform can be implemented as a filter bank decomposition (or filter bank schemes) [23]. Wavelet packet transform (WPT) [6] is a generalization of the discrete wavelet transform (DWT) decomposition which offers a wider range of possibilities for signal representation. To compute this transform it is necesary to select a particular orthogonal basis among all the available basis (or filter banks). Nevertheless, in signal classification applications there is not evidential benefit on working with orthogonal basis. Without this restriction the result of the full WPT decomposition is a highly redundant set of coefficients, which is convenient to optimize in order to use it on signal classification.

* This work is supported by ANPCyT-UNER, under Project PICT No 11-12700, UNL-CAID 012-72 and CONICET.

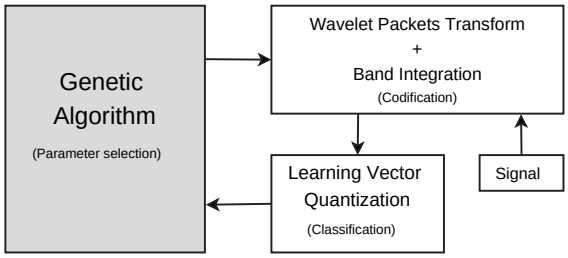

Fig. 1. General scheme of the proposed method.

The projection of a signal of interest in terms of *all* the elements available in the family of wavelet packet basis will be named here as *full wavelet packet decomposition* (FWPD).

Different approaches has been proposed for the optimization wavelet decompositions. In [18] an automatic extraction of high quality features from wavelet coefficients according to signal classification criteria was presented. Another interesting approach is proposed in [8], in which a language based genetic algorithm is used to design wavelet filters that enhance classification performance. A related work, applied to signal approximation is [4], which describes a method to optimize overcomplete decompositions from several dictionaries relying on evolutionary computation techniques. Other works on wavelet decomposition optimization using evolutionary computation are [3,10,21].

We propose a new approach to optimize overcomplete decompositions from several dictionaries, which consists on a parallel algorithm for the genetic FWPD (GA-FWPD) coefficients selection method [24]. In order to measure the goodness of the solutions during the search, the GA uses a learning vector quantization (LVQ) classifier. The scheme of the proposed method is shown in Fig. 1. The methodology, referred to as evolutionary pursuit, relies on evolutionary computation techniques to find a better signal representation.

In the following a brief description of the properties of the WPT, the GAs and parallel computing is presented. Next the proposed method for the selection of the FWPD components and its paralelization are introduced. The last sections present the obtained results and the general conclusions.

1.1 Wavelet and wavelet packet transforms

The interest on wavelet bases is that, in contrast with sine and cosine bases, they are simultaneously localized in time and frequency. This feature is particularly interesting in the case of speech signals which present both stationary and transitory behaviors.

Wavelets can be defined, in a simplified manner, as a function of zero mean, unitary norm and centered in the neighborhood of $t = 0$ [12]:

$$\psi(t) \in L^2(\mathbb{R}); \int_{-\infty}^{\infty} \psi(t)dt = 0; \left\|\psi(t)\right\| = 1. \tag{1}$$

A family of time-frequency atoms is obtained by scaling and translating a wavelet function:

$$\psi_{u,s}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right), \tag{2}$$

with $u, s \in \mathbb{R}$.

In this way the continuous wavelet transform of the signal $x(t)$ is defined as the inner product with this family of atoms:

$$W_x(u,s) = \langle x(t), \psi_{u,s}(t)\rangle = \int_{-\infty}^{+\infty} x(t)\,\psi^*_{u,s}(t)\,dt, \tag{3}$$

The *discrete dyadic wavelet transform* (DWT) of $x[n] \in \mathbb{R}^N$ is obtained by discretizing translation and scaling parameters in (3) as $u = m$ and $s = 2^j$. A fast implementation of DWT based multiresolution analysis exists where the signal is decomposed on details $d_{j+1}[m]$ and approximation $c_{j+1}[m]$ coefficients by a filter bank convolution cascade algorithm. In order to maintain the number of samples, both signals are subsampled to half. Next the aproximation is decomposed again to give a second level of details and approximation, and the process is repeated for each scale j [11]:

$$d_{j+1}[m] = \sum_{n=-\infty}^{\infty} g[n-2m]d_j[n] = d_j * \overline{g}[2m], \tag{4}$$

$$c_{j+1}[m] = \sum_{n=-\infty}^{\infty} h[n-2m]c_j[n] = c_j * \overline{h}[2m], \tag{5}$$

where $g[n]$ and $h[n]$ are the impulse response of the high-pass and low-pass filters associated with the wavelets and scaling functions, $\overline{g}[n] = g[-n]$ and $\overline{h}[n] = h[-n]$.

The WPT could be considered as an extension of the DWT which provides more flexibility on frequency band selection. Using the same reasoning details (high frequency components) can be decomposed as well as approximations (low frequency components). The resulting set of parameters conforms the full WPT decomposition tree.

Wavelet packet analysis allows to represent information in a more flexible time-frequency plane by selecting different sub-trees from the full decomposition. For the selection of the best tree it is possible to make use of the knowledge about signal characteristics and to obtain an efficient representation in the transform domain. For the case of signal compression the criteria is usually based on "entropy" measures, method named as *best orthogonal basis* [1]. Another possibility, closer to the considered problem, is to use the *local discriminant basis* algorithm which provides an appropriate orthogonal basis for signal classification [20]. These criteria are based on the assumption that an orthogonal basis is convenient. Nevertheless, for the case in study there is not evidence on the

sinc(*i*) Laboratory for Signals and Computational Intelligence (http://fich.unl.edu.ar/sinc)
L. D. Vignolo, D. H. Milone, H. L. Rufiner & E. M. Albornoz; "Parallel implementation for wavelet dictionary optimization applied to pattern recognition"
7th Argentine Symposium on Computing Technology. pp. 49-60, 2006.

convenience of a no redundant representation. Because of this, a method which explores a wider range of possibilities is required.

1.2 Genetic algorithms

Genetic algorithms [7] provide the flexibility and robustness required to find satisfactory solutions in complex search spaces [5]. This kind of algorithm also present an implicit parallelism and may be paralleled in a number of ways in order to increase the computational speed [22].

Usually a GA consist on three operators: selection, genetic operation and replacement [22]. The population is conformed by a group of individuals whose information is codified in chromosomes, from which the candidates can be selected for the solution of a problem. Each individual performance is represented by a fitness value which is measured calculating the objective function in a decoded form (phenotype). This function simulates the selective pressure of the environment.

A particular group of individuals (parents) is selected from the population to generate the offspring by the defined genetic operations. Present population is then replaced by the offspring, and several strategies exist to achieve this task. The AG cicle is repeated until a desired termination criterion is reached (for example, a predefined number of generations, a desired fitness value, etc). After the evolution process the best individual of the population is, probably, the desired solution for the problem.

The objective function, which measures the fitness of the individuals, was defined in this case as a classifier based on the *optimized learning vector quantization (O-LVQ)* technique [9]. This classifier uses a set of reference vectors (*codebook*) which are adapted using a set of training patterns in order to represent the distribution of classes. The codebook is initialized with a random selection of training patterns, with the restriction that a specified number of patterns in the nearest neighbors should belong to the same class of the reference vector.

1.3 Parallel computing and message passing

Parallel computing is the use of multiple computers or processors working altogether on the same task. Each processor works on a section of the problem and can interchange information with the others processors. Among the various existing types of parallel computers, *clusters* are the most popular ones. A computer cluster is basically a set of loosely coupled computers that work together by means of high speed local area network. The most popular implementation is a cluster with nodes running Linux OS and free software to implement the parallelism. This configuration is referred to as a Beowulf cluster[1] and is meant to run custom programs which have been designed to exploit the parallelism.

In order to take advantage of this architecture is very important to have the appropiated programming tools, the *message passing* is a widely used programming paradigm for local memory computers. There exist several libraries

[1] http://www.beowulf.org/overview/index.html

							Total
Level	1	2	3	4	5	6	
Nodes	2	4	8	16	32	64	
Groups per band	8	8	4	2	1	1	
Coefficients per level	16	32	32	32	32	64	208

Table 1. Integration scheme applied to the FWPD tree for a 256 sample signal.

for message passing, but they are all based on process comunication by means of messages. The message passing interface (MPI) is the most widely used message passing library and is proposed as a standard by a broadly based committee of vendors, implementors, and users. Its greater advantage is the portability to any kind of computers. This library also allows execution of programs in a transparent way over heterogenous systems, making any necesary data conversion in automatic way and using the correct communication protocol.

2 Materials and methods

This section explains the pre-processing method and the optimization strategy. The first subsection describes the FWPD implementation and the speech corpus used. In the next section the GA-FWPD method and the paralelization strategy are explained.

2.1 Speech corpus and pre-processing

The speech files used for experimentation is a subset of the Albayzin [13] geographic corpus, named Minigeo. This subset is conformed by 600 sound files, with emissions of twelve different speakers (six men and six women) between fifteen and fifty five years old. This data base was phonetically segmented by a Hidden Markov Models based recognition system to obtain the temporal phoneme localization inside each sound file. Also, data is partitioned in three groups, a training data set and a testing set for using during evolution, and a third data set which is used at the end of evolution to validate the classification result.

The sound files were procesed using a WPT implementation which was extended from the DWT algorithms of numerical recipes library [14]. Phoneme signals used for the experiments were 256 samples length, this is 32 mseg segments for 8 kHz sampling frequency. This window size is usual on speech processing and the 256 samples have been centered on the middle point of phonemes.

WPT filters have been used decimating the signal to obtain six decomposition levels, which results on a FWPD tree with 1792 coefficientes. In order to reduce the dimensionality of the search space the coefficients inside each frequency band are integrated by groups. This means that each band is subdivided in groups and then, an energy coefficient is defined by calculating the square of the $L^2 - norm$

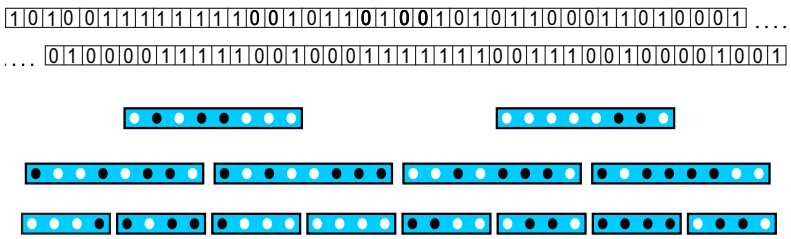

Fig. 2. Codification example with a 80 genes chromosome and the corresponding WPD tree. White dots represent the used coefficients and the black dots represent not used ones.

for each group. Table 1 shows the integration scheme used in this work, which has been chosen heuristically and considering the characteristics of the signal.

The resulting values of band integration are then normalized in order to map all the coefficients to numbers between 0 and 1. This means, if $w_p[k]$ is one of the energy coefficients corresponding to pattern p, then the normalized coefficient will be:

$$\hat{w}_p[k] = \frac{w_p[k]}{\max\limits_{\forall i}\{w_i[k]\}} \tag{6}$$

The corpus files are processed before the evolution starts, so all the FWPD band-integrated and normalized coefficients are saved. Once the coefficients are obtained, they are arranged to conform the train and test files that will be used with the classifier. When a particular individual is evaluated, the classifier uses only the coefficients which are indicated by the respective chromosome.

The implemented binary GA uses roulette wheel selection method and an elitist replacement strategy, which maintains the best individual to the next generation. The genetic operators used in these experiments were classic mutation and one-point crossover.

2.2 Parallel GA-FWPD

The fitness function would have to evaluate the signal representation suggested by a given chromosome, providing a measure of the phoneme class separability obtained with it. The proposed fitness function consists of a phoneme classifier and the recognition rate is the fitness value for each evaluated individual. This classifier is based in the LVQ method, and was chosen, mainly, due to its low computational cost [9].

Once the data base is processed, each pattern (representing a phoneme) is composed by the normalized and band-integrated FWPD coefficients. These patterns are labeled and could be used to train and test the O-LVQ based classifier. From all these coefficients, a particular subset is chosen by the chromosome of a given individual before its evaluation.

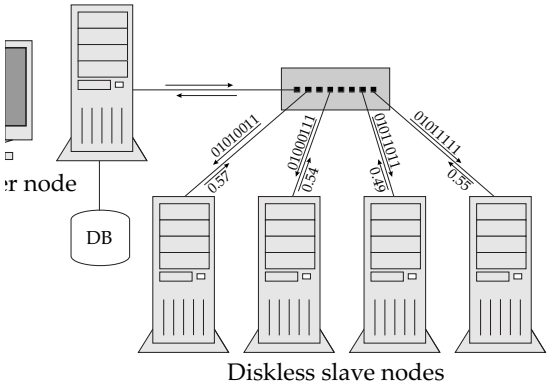

Fig. 3. GA paralelization strategy.

A simple evolution model with binary chromosomes has been used. Each individual represents a different selection of the FWPD band-integrated coefficients. Each gene in a chromosome represents one of the 208 coefficients, and its value indicates whether that coefficient is used or not to codify the signals (Figure 2). The GA initialization consists on filling the genes randomly, as the codification could be redundant and no restriction is imposed for coefficients combination.

The selection of individuals should be done considering the set of coefficients represented by each chromosome. The selection process should assign more probablity to the chromosomes which codify the best signal representations and these will be those that allow better classification results.

This GA needs to evaluate each individual training and testing a phoneme classifier. This implies that, to reach a good solution, it is precise to make many generations which implies a great computacional cost. That is to say, for each experiment 500 generations are needed, for each generation is necessary to evaluate 100 individuals, for each individual there are at least 10 LVQ epochs of 2500 training patterns and 500 test patterns. In order to do this in a reasonable time the parallel execution is necessary.

The chosen paralelization follows a *master-slave* strategy and was implemented using MPICH2 library[2], which is one of the most widely used MPI implementations. The master program is the GA itself, this means that it does the genetic operations and individuals selection during evolution. The slave program makes use of the O-LVQ based classifier routine and does the most costly work, this is, to obtain the fitness value for each individual (Figure 3). This strategy is efficient because the slave nodes do not need to communicate between them, but only with the master node. More over, the comunication with the master node consists on the chromosome and the fitness value only.

[2] http://www-unix.mcs.anl.gov/mpi/mpich2/

3 Results and discusion

In previous works, various wavelet families have been tested in order to find which one is the most convenient for signal classification [19]. In this work the tests included the most widely used families, between which we can mention Meyer, Daubechies, Symmlets, Coiflets y Splines [2]. As result, the 4th order Coiflet family was chosen to be used on the following experiments.

The experiments include the phonemes /a/, /e/, /i/, /o/, /u/, /b/, /d/, /p/ and /t/ from the Spanish. The five vowels were included because of their obvious importance in the language, while the last four phonemes are classified as occlusive phonemes [15] and they have been included beause of their similar characteristics that make them particularly difficult to distinguish.

For the first experiment a codebook of 117 vectors (13 per phoneme) was used with in the classifier and the initial learning rate for each one of this vectors was set on 0.02 as this value allows stabilization of the codebook vectors. The classifier training was made in 6 epochs with 1449 patterns while 252 patterns were left for testing. For the GA, the population size was set on 100 individuals while crossover and mutation probabilities were set on 0.9 and 0.05 respectively. The best solution was found after 26 generations (from the total of 255) and gave 57.94% of correct classifications as result.

The codebook initialization has a random component and, consequently, the training with a fixed set of parameters does not result allways on the same reference vectors. This means that repeated evaluations for a given individual may not result exactly on the same fitness value. In order to obtain a better estimation of the classification rate, after evolution, the best individual found was evaluated ten times. The training and testing were repeated with a set of test patterns which were not used during the evolution. This data partition consists on 2637 training patterns and 450 test patterns. As result, an average of 53.69% correct classifications with a standard deviation of 2.33%. This result shows that the initialization presents a problem, as the validation average is smaller than the result obtained during evolution. It is also important to consider that this deviation on fitness computation may probably affect the evolution.

In order to analyze the formulated problem, the next experiment consisted on eliminating the randomness of the codebook initialization in the classifier. The stated hypothesis consists in that all individuals are evaluated exactly the same way, the GA will be able to find a better solution. After the first 216 generations (from a total of 500) the GA found the individual which gave a result of 57.78% success. In this case, the same validation procedure with ten different initilizations, gave a resulting average of 56.67% with a standard deviation of 2.90%.

Although an improvement was obtained using the mentioned strategy, it is also possible that the evolution was oriented to optimize the classification based on a fixed O-LVQ initialization. This means, as the codebook initialization is not random, then it is possible that the individuals of the GA are adapted to its sequential order and the found solution is good only when this initialization sequence is used.

%	/e/	/t/	/p/	/o/	/u/	/i/	/d/	/b/	/a/	Average
/e/	76.06	02.12	03.64	05.15	03.64	01.82	01.51	03.33	02.73	
/t/		31.82	61.51	00.30			01.82	04.54		
/p/		16.36	78.18				02.42	03.03		
/o/	10.61		00.61	42.42	14.24		02.73	04.24	25.15	
/u/		01.82	05.15	08.48	59.39	01.51	00.61	14.24	08.79	
/i/	08.18	00.91	00.30		00.30	86.97	00.61	02.73		
/d/	31.82	04.24	17.27	07.57	04.54	09.09	10.61	04.54	10.30	
/b/	02.42	01.82	13.03	04.54	09.70		06.06	62.12	00.30	
/a/	00.30		00.30	11.82	01.21		00.30	01.21	84.85	
Average										59.16

Table 2. Confusion matrix obtained after validation for the last experiment.

In this sense a new strategy was stated, which consists on changing the initialization sequence on each generation, but not for each individual of the same generation. In this case, when the individuals of a given generation are evaluated, the codebook vectors are initialized choosing the train patterns in the same order. In this experiment the best individual found gave the result of 67.78% classification success, while the validation process obtained an average of 53.33% with a standard deviation of 2.80%.

Finally, the same experiment was repeated but using a generational gap of 10 individuals, in addition to the elitist strategy of the GA. Which implies that 10 individuals are selected randomly from the parents of a population and maintained without changes to the next generation. The objective of this strategy was to maintain extra information from all generations, allowing individuals evaluated using different initialization sequences to coexist in the same generation. In this case the best solution gave a result of 64.07% succesfull classifications, and the GA converged in 355 generations. Figure 4 shows the fitness value against the number of generations for this last experiment. By means of the validation process, again, a lower success percentage was obtained. The average was 59.16% successfull classifications with a standard deviation of 2.91%. Table 2 shows a confusion matrix obtained as average from validation results. As this matrix shows, the /t/ phoneme is generally (61.51%) classified as /p/. This error turn up because the experimental data was taken from the central part of the samples and the plosive phonemes, like /p/ and /t/, have their most particular attributes on the beginning (the phoneme plosion). A solution could be to distinguish the plosive phonemes using their context. In the /d/ phoneme case, the problem is similar.

In order to contrast results, experiments were made parameterizing the signals with the DWT, considering the same phonetic group and the same classifier parameters. As Table 3 shows, the parameterization found using GA-FWPD method makes possible to obtain better classification results.

sinc(*i*) Laboratory for Signals and Computational Intelligence (http://fich.unl.edu.ar/sinc)
L. D. Vignolo, D. H. Milone, H. L. Rufiner & E. M. Albornoz; "Parallel implementation for wavelet dictionary optimization applied to pattern recognition"
7th Argentine Symposium on Computing Technology. pp. 49-60, 2006.

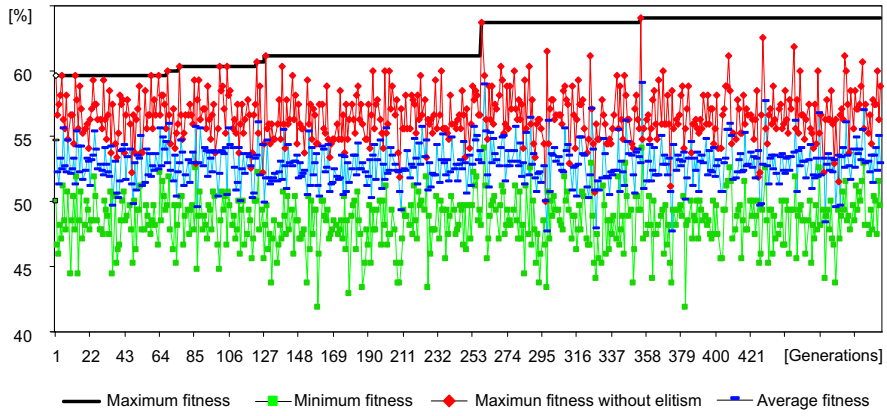

Fig. 4. Classification success percentage against the generations number.

Phoneme	/a/	/e/	/i/	/o/	/u/	/b/	/d/	/p/	/t/	Average
GA-FWPD	84.85	76.06	86.97	42.42	59.39	62.12	10.61	78.18	31.82	59.16
DWT	69.39	54.54	84.54	54.54	31.51	58.48	59.69	4.85	31.21	49.86

Table 3. Comparison of the results using the GA-FWPD method against DWT parameterization. Success classification percentages for each phoneme.

Note that each experiment gave a very different representation as result, which do not a present a common structure and do not allow a simple graphical analysis.

The tests with the parallel algorithm were done in a Beowulf cluster with eleven computers[3] and the tests with the sequential algorithm in a single cluster node. All of them have Pentium IV processors with a clock frequency of 3 GHz.

In the cluster, one computer ran the master program which control the evolution and the others ran a program to calculate the individual fitness value.

The computational cost vary with the amount of used patterns, the number of epochs and the codebook size used for individuals evaluation, in addition to the population size and the number of generations. In order to do a processing time comparison, parameters from the first experiment were taken as an example.

To evaluate 255 generations the required time is 19380 seconds in the cluster case and 191999 seconds in the single node case. Then, the cluster spends 76 seconds for each generation, as long as one processor spends 753 seconds.

Eleven processors were used but the master node load is insignificant in comparison with the slave nodes load, due to this, the measures was taking for 10 processors. Speedup is a measure to know how faster is a parallel algorithm than the corresponding sequential algorithm and is defined as $S_p = T_1/T_p$. In this case it was $S_{10} = 9.94$ and makes evident the parallel algorithm significance.

[3] We thanks to CIMEC for the access to the cluster for this experiment

sinc(*i*) Laboratory for Signals and Computational Intelligence (http://fich.unl.edu.ar/sinc)
L. D. Vignolo, D. H. Milone, H. L. Rufiner & E. M. Albornoz; "Parallel implementation for wavelet dictionary optimization applied to pattern recognition"
7th Argentine Symposium on Computing Technology. pp. 49-60, 2006.

Other useful measure is the *eficiency*, defined as S_p/p. Its value express a relation between the real computation time and the lost time because of comunication and sincronization. Using the parallel and the sequential implementation for the same experiment, the efficiency was 0.99 over 10 processors. Strictly, the number of used processors was eleven, so the efficiency is at least 0.9.

The used cluster (Figure 3) has disk-less slave nodes which uses the linux net file system (NFS). The communication between master and slaves programs is minimum, but the slaves have to fetch the trainning data from the master node and this produce a delay. The efficiency of the implemented parallelization is very close to the ideal value, this is possible because the improved NFS management allows slave nodes to keep a copy of the training data in their own memory so the network transfers are carried out only once for each node.

4 Conclusion and future works

This paper presented a technique that allow to choose the relevant components from the FWPD for signal classification. In this way, it is possible to find the "best" representation to feed a speech recognition automatic system, like an alternative from the based on DWT.

The experiments have shown that the used initialization process introduced variance in the results. Some alternatives have been tested to fix this problem and the GA has been able to find acceptable solutions.

The experimental tests show an important time reduction owing to the genetic algorithm parallel implementation. This make possible to use more robust classification methods in the future.

Although, the plosive phonemes were not correctly classified, the method is able to distinguish them as a group. A possible solution to this problem could be to take more, or different, samples from the phoneme signals.

A future work is to procure a general conclusion about structure of the best representation obtained and look for a relation with the characteristics of the used signal. Others future works could be the utilization of the best representation in a Hidden Markov Models based automatic speech recognition system and to incorporate signals with noise in the cross validation tests.

References

1. R. Coifman and M. V. Wickerhauser. Entropy-based algorithms for best basis selection. *IEEE Transactions on Information Theory*, 38(2):713–718, 1992.
2. I. Daubechies. *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992.
3. A. R Ferreira da Silva. Evolutionary-based methods for adaptive signal representation. *Signal Processing*, 81:927–944, 2001.
4. A. R Ferreira da Silva. Approximations with evolutionary pursuit. *Signal Processing*, 83(3):465–481, 2003.
5. D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

6. N. Hess-Nielsen and M.V. Wickerhouser. Wavelets and time-frequency analisys. *Proceedings of the IEEE*, 84(4):523–540, 1996.
7. J.H. Holland. *Adaptation in Natural and Artificial System.* University of Michigan Press, Ann Arbor, 1975.
8. E. Jones, P. Runkle, N. Dasgupta, L. Couchman, and L. Carin. Genetic algorithm wavelet design for signal classification. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(8):890–895, 2001.
9. T. Kohonen. Improved versions of learning vector quantization. In *Proc. of the Int. Joint Conf. on Neural Networks*, pages 545–550, San Diego, Junio 1990.
10. M. M. Lankhorst, M. D. Van der Laan, and W. A. Halang. Wavelet-based signal approximation with genetic algorithms. *Systems Analysis Modelling Simulation*, 43(11):1503–1528, 2003.
11. S.G. Mallat. A theory of multiresolution of signal decomposition: the wavelet representation. *IEEE Trans. Pattern Anal. Machine Intell.*, 11(7):674–693, 1989.
12. S.G. Mallat. *A Wavelet Tour of signal Processing.* Academic Press, second edition, 1999.
13. A. Moreno, D. Poch, A. Bonafonte, E. Lleida, J. Llisterri, J. Mari, and C. Nadeu. Albayzin speech database design of the phonetic corpus. Technical report, Universitat Politnica de Catalunya (UPC), Dpto. DTSC, 1993.
14. W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C, The Art of Scientific Computing.* Cambridge University Press, second edition, 1992.
15. A. Quilis. *Tratado de Fonología y Fonética Españolas.* Biblioteca Románica Hispánica. Editorial Gredos, Madrid, 1993.
16. L.R. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition.* Prentice Hall, NJ, 1993.
17. L.R. Rabiner and R.W. Schafer. *Digital Processing of Speech Signals.* Prentice Hall, NJ, 1978.
18. S. Ray and A. Chan. Automatic feature extraction from wavelet coefficients using genetic algorithms. In *Proceedings of the 2001 IEEE Signal Processing Society Workshop*, pages 233–241. Neural Networks for Signal Processing XI, Sept. 2001.
19. H. Rufiner. Comparación entre análisis onditas y fourier aplicados al reconocimiento automático del habla. Master's thesis, Universidad Autónoma Metropolitana, Iztapalapa, 1996.
20. N. Saito. *Local feature extraction and its applications using a library of bases.* PhD thesis, Yale University, New Haven, USA, 1994. Director-Ronald R. Coifman.
21. T. Schell and A Uhl. Optimization and assessment of wavelet packet decompositions with evolutionary computation. *EURASIP Journal on Applied Signal Processing*, (8):806–813, 2003.
22. K.S. Tang, K.F. Man, S. Kwong, and Q. He. Genetic algorithms and their applications. *IEEE Signal Processing*, 13(6):22–29, 1996.
23. M. Vetterli and C. Herley. Wavelets and filter banks: Theory and design. *IEEE Trans. Signal Proc.*, 40(10):2207–2232, 1992.
24. L.D. Vignolo and D.H. Milone. Optimización de paquetes de onditas para la clasificación de señales. In *Anales de la XI Reunión de Trabajo en Procesamiento de la Información y Control*, pages 57–62, Río Cuarto (Córdoba), Setiembre 2005.