12

Optimal Feature Generation with Genetic Algorithms and FLDR in a Restricted-Vocabulary Speech Recognition System

Julio César Martínez-Romo[1], Francisco Javier Luna-Rosas[2], Miguel Mora-González[3], Carlos Alejandro de Luna-Ortega[4] and Valentín López-Rivas[5]
[1,2,5]*Instituto Tecnológico de Aguascalientes*
[3]*Universidad de Guadalajara, Centro Universitario de los Lagos*
[4]*Universidad Politécnica de Aguascalientes*
Mexico

1. Introduction

In every pattern recognition problem there exist the need for variable and feature selection and, in many cases, feature generation. In pattern recognition, the term variable is usually understood as the raw measurements or raw values taken from the subjects to be classified, while the term feature is used to refer to the result of the transformations applied to the variables in order to transform them into another domain or space, in which a bigger discriminant capability of the new calculated features is expected; a very popular cases of feature generation are the use of principal component analysis (PCA), in which the variables are projected into a lower dimensional space in which the new features can be used to visualize the underlying class distributions in the original data [1], or the Fourier Transform, in which a few of its coefficients can represent new features [2], [3]. Sometimes, the literature does not make any distinction between variables and features, using them indistinctly [4], [5].

Although many variables and features can be obtained for classification, not all of them posse discriminant capabilities; moreover, some of them could cause confusion to a classifier. That is the reason why the designer of the classification system will require to refine his choice of variables and features. Several specific techniques for such a purpose are available [1], and some of them will be reviewed later on in this chapter.

Optimal feature generation is the generation of the features under some optimality criterion, usually embodied by a cost function to search the solutions' space of the problem at hand and providing the best option to the classification problem. Examples of techniques like these are the genetic algorithms [6] and the simulated annealing [1]. In particular, genetic algorithms are used in this work.

Speech recognition has been a topic of high interest in the research arena of the pattern recognition community since the beginnings of the current computation age [7], [8]; it is due, partly, to the fact that it is capable of enabling many practical applications in artificial intelligence, such as natural language understanding [9], man-machine interfaces, help for the impaired, and others; on the other hand, it is an intriguing intellectual challenge in which new mathematical methods for feature generation and new and more sophisticated classifiers appear nearly every year [10], [11]. Practical problems that arise in the implementation of speech recognition algorithms include real-time requirements, to lower the computational complexity of the algorithms, and noise cancelation in general or specific environments [12]. Speech recognition can be user or not-user dependant.

A specific case of speech recognition is word recognition, aimed at recognizing isolated words from a continuous speech signal; it find applications in system commanding as in wheelchairs, TV sets, industrial machinery, computers, cell phones, toys, and many others. A particularity of this specific speech processing niche is that usually the vocabulary is comprised of a relatively low amount of words; for instance, see [13] and [14].

In this chapter we present an innovative method for the restricted-vocabulary speech recognition problem in which a genetic algorithm is used to optimally generate the design parameters of a set of bank filters by searching in the frequency domain for a specific set of sub-bands and using the Fisher's linear discriminant ratio as the class separability criterion in the features space. In this way we use genetic algorithms to create optimum feature spaces in which the patterns from **N** classes will be distributed in distant and compact clusters. In our context, each class $\{\omega_0, \omega_1, \omega_2, \ldots, \omega_{N-1}\}$ represents one word of the lexicon. Another important part of this work is that the algorithm is required to run in real time on dedicated hardware, not necessarily a personal computer or similar platform, so the algorithm developed should has low computational requirements.

This chapter is organized as follows: the section 2 will present the main ideas behind the concepts of variable and feature selection; section 3 presents an overview of the most representative speech recognition methods. The section 4 is devoted to explain some of the mathematical foundations of our method, including the Fourier Transform, the Fisher's linear discriminant ratio and the Parseval's theorem. Section 5 shows our algorithmic foundations, namely the genetic algorithms and the backpropagation neural networks, a powerful classifier used here for performance comparison purposes. The implementation of our speech recognition approach is depicted in section 6 and, finally, the conclusions and the future work are drawn in section 7.

2. Optimal variable and feature selection

Feature selection refers to the problem of selecting features that are most predictive of a given outcome. Optimal feature generation, however, refers to the derivation of features from input variables that are optimal in terms of class separability in the feature space. Optimal feature generation is of particular relevance to pattern recognition problems because it is the basis for achieving high correct classification rates: the better the discriminant features are represented, the better the classifier will categorize new incoming patterns. Feature generation is responsible for the way the patterns lay in the features space, therefore, shaping the decision boundary of every pattern recognition problem; linear as well as non-linear classifiers can be beneficiaries of well-shaped feature spaces.

The recent apparition of new and robust classifiers such as support vector machines (SVM), optimum margin classifiers and relevance vector machines [4], and other robust kernel classifiers seems to demonstrate that the new developments are directed towards classifiers which, although powerful, must be preceded by reliable feature generation techniques. In some cases, the classifiers use a filter that consists of a stage of feature selection, like in the Recursive Feature Elimination Support Vector Machine [15], which eliminates features in a recursive manner, similar to the backward/forward variable selection methods [1].

2.1 Methods for variable and feature selection and generation

The methods for variable and feature selection are based on two approaches: the first is to consider the features as scalars -*scalar feature selection*-, and the other is to consider the features as vectors –*feature vector selection*-. In both approaches a class separability measurement criteria must be adopted; some criteria include the receiver operating curve (ROC), the Fisher Discriminant Ratio (FDR) or the one-dimensional divergence [1]. The goal is to select a subset of *k* from a total of *K* variables or features. In the sequel, the term features is used to represent variables and features.

2.1.1 Scalar feature selection

The first step is to choose a class separability measuring criterion, C(K). The value of the criterion C(K) is computed for each of the available features, then the features are ranked in descending order of the values of C(K). The *k* features corresponding to the *k* best C(K) values are selected to form the feature vector. This approach is simple but it does not take into consideration existing correlations between features.

2.1.2 Vector feature selection

The scalar feature selection may not be effective with features with high mutual correlation; another disadvantage is that if one wishes to verify all possible combinations of the features –in the spirit of optimality- then it is evident that the computational burden is a major limitating factor. In order to reduce the complexity some suboptimal procedures have been suggested [1]:

Sequential Backward Selection. The following steps comprise this method:

a. Select a class separability criterion, and compute its value for the feature vector of all the features.
b. Eliminate one feature and for each possible combination of the remaining features recalculate the corresponding criterion value. Select the combination with the best value.
c. From the selected *K-1* feature vector eliminate one feature and for each of the resulting combinations compute the criterion value and select the one with the best value.
d. Continue until the feature vector consists of only the *k* features, where *k* is the predefined size.

The number of computations can be calculated from: $1+1/2\,((K+1)K - k(k+1))$.

Sequential Forward Selection. The reverse of the previous method is as follows:

a. Compute the criterion value for each individual feature; select the feature with the "best" value,
b. From all possible two-dimensional vectors that contains the winner from the previous step. Compute the criterion value for each of them and select the best one.

2.1.3 Floating search methods

The methods explained suffer from the *nesting* effect, which means that once a feature (or variable) has been discarded it can't be reconsidered again. Or, on the other hand, once a feature (or variable) was chosen, it can't be discarded. To overcome these problems, a technique known as *floating search method,* was introduced by Pudin and others in 1994 [1], allowing the features to enter and leave the set of the *k* chosen features. There are two ways to implement this technique: one springs from the forward selection and the other from de backward selection rationale. A three steps procedure is used, namely *inclusion*, *test*, and *exclusion.* Details of the implementation can be found in [1], [16].

2.1.4 Some trends in feature selection

Recent work in feature selection are, for instance, the one of Somol *et al*., [17], where besides of optimally selecting a subset of features, the size of the subset is also optimally selected. Sun and others [18] faced the problem of feature selection in conditions of a huge number or irrelevant features, using machine learning and numerical analysis methods without making any assumptions about the underlying data distributions. In other works, the a feature selection technique is accompanied by instance selection; instance selection refers to the "orthogonal version of the problem of feature selection" [19], involving the discovery of a subset of instances that will provide the classifier with a better predictive accuracy than using the entire set of instances in each class.

2.2 Optimal feature generation

As can be seen from section 2.1, the class separability measuring criterion in feature selection is used just to measure the effectiveness of the *k* features chosen out of a total of K features, with independence of how the features were generated. The topic of optimal feature generation refers to involving the class separability criterion as an integral part of the feature generation process itself. The task can be expressed as: If x is an m-dimensional vector of measurement samples, transform it into another l-dimensional vector andso that some class separability criterion is optimized. Consider, to this end, the linear transformation $y=A^T x$.

By now, it will suffice to note the difference between *feature selection* and *feature generation*.

3. Speech recognition

For speech processing, the electrical signal obtained from an electromechanoacoustic transducer is digitized and quantized at a fixed rate (the sampling frequency, F_s), and subsequently segmented into small frames of a typical duration of 10 milliseconds. Regarding to section 2, the raw digitized values of the voice signal will be considered here as the input variable; the mathematical transformations that will be applied to this variable will produce the *features*.

Two important and widely used techniques for speech recognition will be presented in this section due to its relevance to this field. *Linear Predictive Coding*, or *LPC*, is a predictive technique in which a linear combination of some *K* coefficients and the last *K* samples from the signal will predict the value of the next one; the *K* coefficients will represent the distinctive features. The following section will explain the LPC method in detail.

3.1 Linear Predictive Coding (LPC)

LPC is one of the most advanced analytical techniques used in the estimation of patterns, based on the idea that the present sample can be predicted from a linear combination of some past samples, generating a spectral description based on short segments of signal considering a signal $s[n]$ to be a response of an all-pole filter excitation $u[n]$.

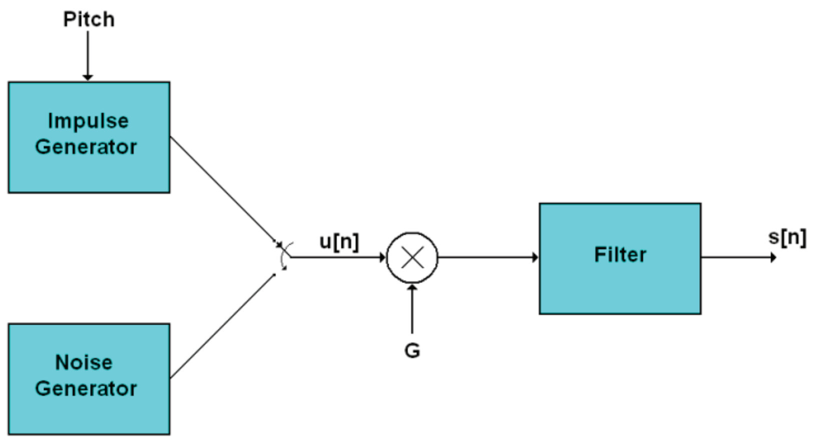


Fig. 1. LPC model of speech.

Figure 1 shows the model the LPC is based on, considering that the excitation $u[n]$ is the pattern waiting to be recognized. The transfer function of the filter is described as [3]:

$$H(z) = \frac{S(z)}{U(Z)} = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}} = \frac{G}{A(z)}, \tag{1}$$

where G is a gain parameter, $\boldsymbol{a_k}$ are the coefficients of filter and $\boldsymbol{p}$ determines the order of the filter. In Figure 1, the samples $s[n]$ are related to the excitation $u[n]$ by the equation:

$$s[n] = \sum_{k=1}^{p} a_k s[n-k] + Gu[n], \tag{2}$$

Considering that the linear combination of past samples is calculated by using an estimator $\tilde{s}[n]$ which is denoted by:

$$\tilde{s}[n] = \sum_{k=1}^{p} a_k s[n-k], \tag{3}$$

the error in the prediction is determined by the lack of accuracy with respect to $s[n]$, which is defined as [20]:

$$e[n] = s[n] - \tilde{s}[n] = s[n] - \sum_{k=1}^{p} a_k s[n-k], \tag{4}$$

$$e[z] = s[z]\left(1 - \sum_{k=1}^{p} a_k z^{-k}\right). \tag{5}$$

from equation (5), it is possible to recognize that the sequence of prediction of the error has in its components a FIR-type filter system which is defined by:

$$A(z) = 1 - \sum_{k=1}^{p} a_k z^{-k} = \frac{E(z)}{S(z)}, \tag{6}$$

equations (2) and (4) show that $e[n]=Gu[n]$. The estimation of the prediction coefficients is obtained by minimizing the error in the prediction. Where $e[n]^2$ denotes the square error of the prediction and E is the total error over a time interval (m). The prediction error in a short time segment is defined as:

$$E = \sum_{m} e[m]^2 = \sum_{m} (s[m] - \sum_{k=1}^{p} a_k s[m-k])^2, \tag{7}$$

the coefficients $\{a_k\}$ minimize the prediction of error E on the fragment obtained by the partial derivatives of E with respect to such coefficients; this means that:

$$\frac{\partial E}{\partial a_i} = 0, \quad 1 \le i \le p. \tag{8}$$

Through equations (7) and (8) the final equation is obtained:

$$\sum_{k=1}^{p} a_k \sum_{n} s[n-k]s[n-i] = -\sum_{n} s[n]s[n-i], \qquad 1 \le i \le p, \tag{9}$$

this equation is written in terms of least squares and is known as a normal equation. For any definitions of the signal $s[n]$, equation (9) forms a set of $\boldsymbol{p}$ equations with $\boldsymbol{p}$ unknowns that must be solved for coefficients $\{a_k\}$, trying to reduce the error E of equation (7). The minimum total squared error, denoted by Ep, is obtained by expanding equation (7) and substituting the result in equation (9), this is:

$$Ep = \sum_{n} s^2[n] + \sum_{k=1}^{p} a_k \sum_{n} s[n]s[n-k], \tag{10}$$

using the autocorrelation method to solve it [8].

For application, it is assumed that the error of equation (7) is minimized for infinite duration defined as $-\infty < n < \infty$, thus equations (9) and (10) are simplified as:

$$\sum_{k=1}^{p} a_k R(i-k) = -R(i), \qquad 1 \le i \le p, \tag{11}$$

$$Ep = R(0) + \sum_{k=1}^{p} a_k R(k), \tag{12}$$

where:

$$R(i) = \sum_{n=-\infty}^{\infty} s[n]s[n+i], \tag{13}$$

which is the autocorrelation function of the signal $s[n]$, with $R(i)$ as an even function. The coefficients $R(i\text{-}k)$ generate auto-correlation matrix, which is a symmetric Toeplitz matrix; ie, all elements in each diagonal are equal. For practical purposes, the signal $s[n]$ is analyzed in a finite interval. One popular method of approach this is by multiplying the signal $s[n]$ times a window function $w[n]$ in order to obtain an $s'[n]$ signal:

$$s'[n] = \begin{cases} s[n]w[n], \; 0 \le n \le N-1 \\ 0, \qquad \text{otherwise}' \end{cases} \tag{14}$$

Using equation 14, the auto-correlation function is given by:

$$R(i) = \sum_{n=0}^{N-1-i} s'[n]s'[n+i], \qquad i \ge 0. \tag{15}$$

One of the most common ways to find the coefficients $\{a_k\}$, is by computational methods, where the equation (11) is expanded to a matrix with the form:

$$\begin{bmatrix} R_0 & R_1 & R_2 & \cdots & R_{p-1} \\ R_1 & R_0 & R_1 & \cdots & R_{p-2} \\ R_2 & R_1 & R_0 & \cdots & R_{p-3} \\ \vdots & \vdots & \vdots & & \vdots \\ R_{p-1} & R_{p-2} & R_{p-3} & \cdots & R_0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \\ \vdots \\ R_p \end{bmatrix}, \tag{16}$$

and it is necessary to use an algorithm to find these coefficients; one of the most commonly used, is the Levinson-Durbin one, which is described below [21]:

```
$E^0 = R(0)$
$a_0 = 1$
for $i = 1, 2, \ldots, p$
        $k_i = \dfrac{\left( R(i) - \sum_{j=1}^{i-1} a_j^{(i-1)} R(i-j) \right)}{E^{(i-1)}}$
$a_i^{(i)} = k_i$
```

```
if $i > 1$ then for $j = 1,2,...,i-1$
        $a_j^{(i)} = a_j^{(i-1)} - k_i a_{i-j}^{(i-1)}$
end
$E^{(i)} = (1-k_i^2)E^{(i-1)}$
end
$a_j = a_j^{(p)} \quad j = 1,2,...,p.$
```

An important feature of this algorithm is that, when making the recursion, an estimation of the half-quadratic prediction error must be made. This prediction satisfies the system function given in equation (17), which corresponds to the term A (z) of equation (1); namely:

$$A^{(i)}(z) = A^{(i-1)}(z) - k_i z^{-i} A^{(i-1)} z^{-1}, \tag{17}$$

where the fundamental part for the characterization of the signal in coefficients of prediction, is met by establishing an adequate number of coefficients ***p*,** according to the sampling frequency (fs) and based on the resonance in kHz [3] which is:

$$p = 4 + \frac{f_s}{1000} \tag{18}$$

where the optimal number of LPC coefficients is the one that represents the lowest mean square error possible. Figure 2 shows the calculation of LPC in a voice signal with 8 kHz sampling rate and the effect of varying the number of coefficients in a segment (frame).

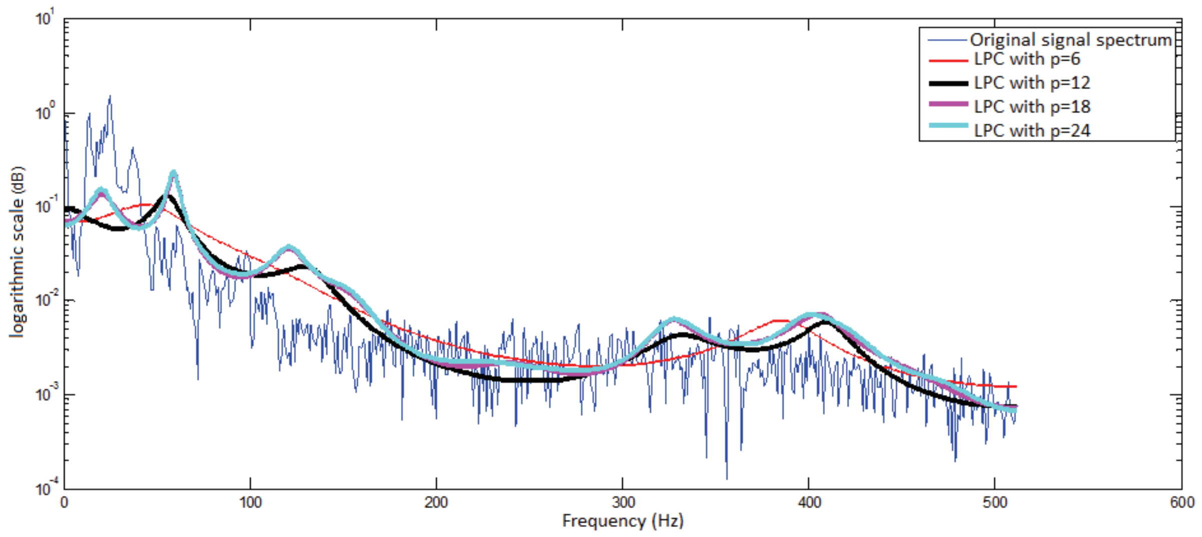


Fig. 2. Comparison of the original signal spectrum and LPC envelope with different numbers of coefficients.

3.2 Dynamic Time Warping (DTW)

Another commonly used technique used in speech recognition is the dynamic time warping. Is presented here, again, for it relevance to the this field. Dynamic time warping is a technique widely used in pattern recognition, particularly oriented to temporal distortions between vectors, such as the time of writing, speed of video camera, the omission of a letter, etc. These temporal variations are not proportional and vary accordingly to each person, object or event, and those situations are not repetitive in any aspects. DTW uses *dynamic programming* to find similarities and differences between two or more vectors.

This method considers two sequences representing feature vectors defined by $a(i)$, i=1,2,…,I and $b(j)$, j=1,2,…,J, where, in general, the number of elements differs in each vector (I≠J). The aim of DTW is to find an appropriate distance between the two sequences and in a two-dimensional plane, where each sequence represents one axis, and each point corresponds to the local relationship between two sequences. The nodes (i,j) of the plane are associated with a cost that is defined by the function $d(c)=d(i,j)=|a(i)-b(j)|$, which represents the distance between the elements $a(i)$ and $b(j)$.

The collection of points begins in the starting point (i_0,j_0) and finishes in the (i_k,j_k) nodes, and it being an ordered pair of size k, where k is the number of nodes along the way. Every path established with the points is associated with a total cost D and defined by

$$D = \sum_{k=0}^{K-1} d(i_k, j_k), \tag{19}$$

and the distance between the two sequences is defined as the minimum value D of all the possible paths

$$D(a,b) = \min_k(D). \tag{20}$$

There are normalization and temporary limitations in the search for the minimum distance between patterns to compare [22], [1]. These limitations are: endpoint, monotonicity conditions, local continuity, global path and slope weight.

The final point is bounded by the size of windowing and performed in each pattern, at most cases is empirical and defined to extremes, that is:

$$\begin{aligned} i(1) = 1, j(1) = 1 \\ i(K) = I, j(K) = J. \end{aligned} \tag{21}$$

Figure 3 shows an example in which it is only partially considered one of the sequences, a situation that is not allowed to search the minimum cost.

The monotonicity conditions try to maintain the temporal order of the normalization of the time, and avoid negative slopes, by

$$i_{k+1} \geq i_k \tag{22}$$

and

$$j_{k+1} \geq j_k. \tag{23}$$

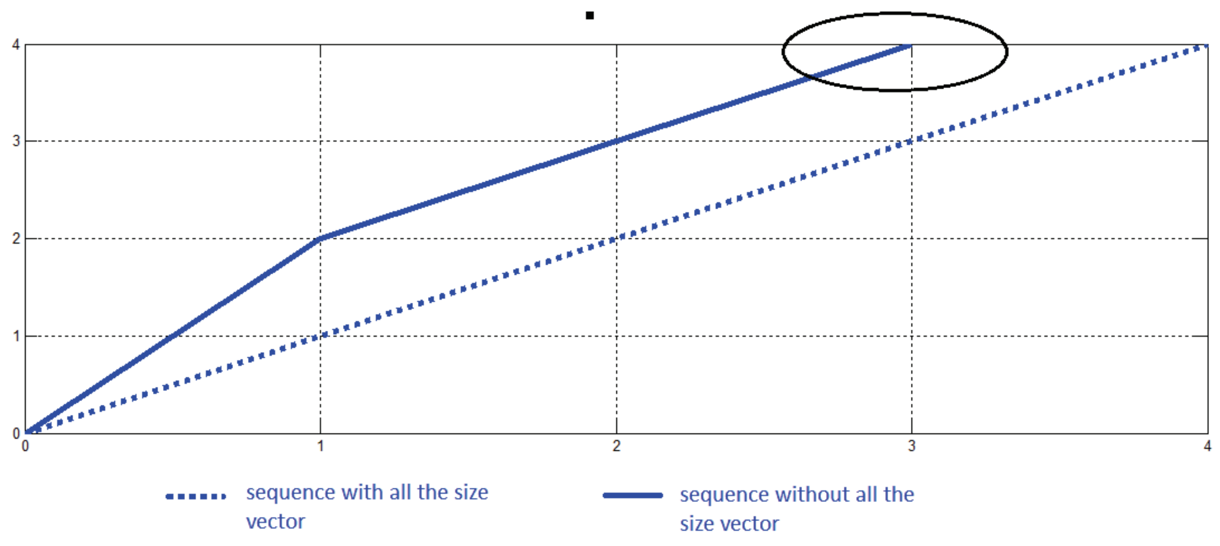


Fig. 3. Example of a sequence that violates the rule of the endpoint.

Figure 4 shows an example without monotonicity paths, which are not allowed to find the optimal path.

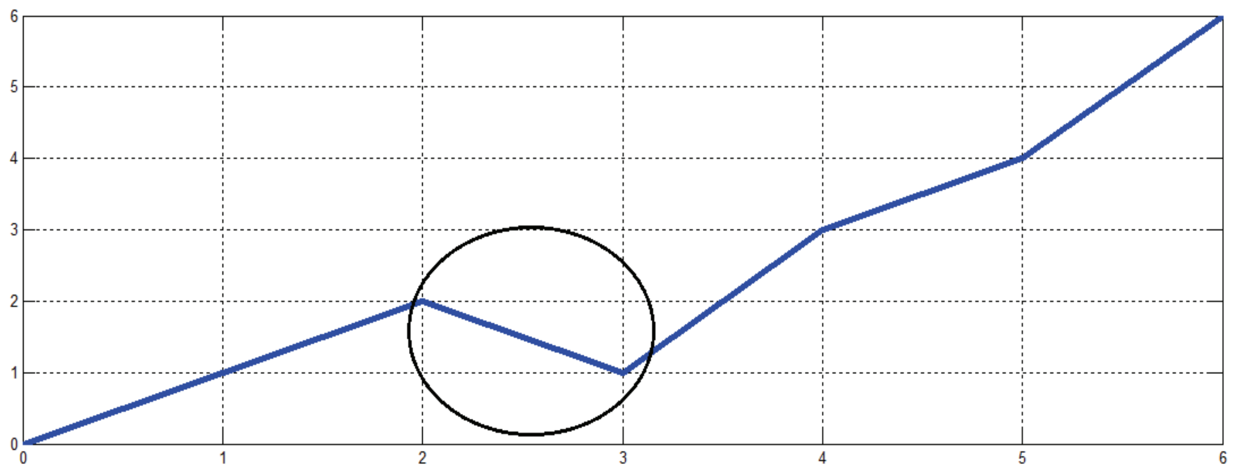


Fig. 4. Without monotonicity path example.

The continuity conditions

$$i(K) - i(k-1) \leq 1 \quad \text{And} \quad j(K) - j(K-1) \leq 1, \tag{24}$$

are defined by maintaining the relationship between two consecutive points of the form:

$$c(K-1) = \begin{cases} (i(k), j(k)-1), \\ (i(k)-1, j(k)-1), \\ (i(k)-1, j(k)), \end{cases} \tag{25}$$

Global limitations define a region of nodes where the optimal path is found, and is based on a parallelogram that offers a feasible region [7], thereby avoiding unnecessary regions involved in processing. Figure 5 shows the values of the key points of the parallelogram.

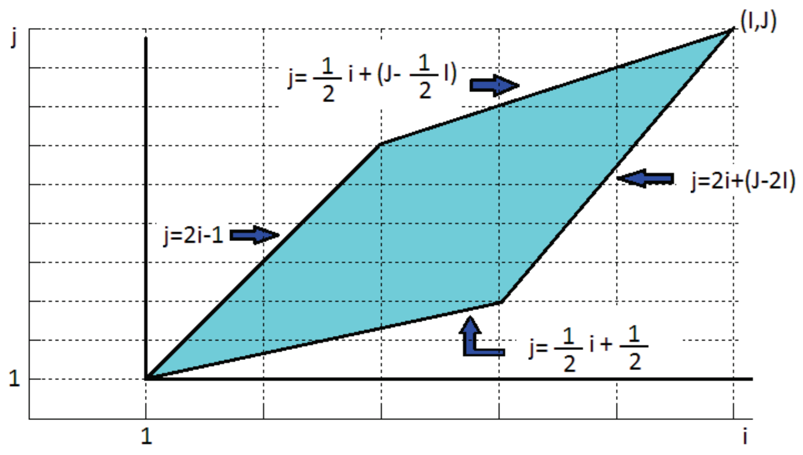

j
1
1
i
(I,J)
$j=\frac{1}{2}i+(J-\frac{1}{2}I)$
$j=2i+(J-2I)$
$j=2i-1$
$j=\frac{1}{2}i+\frac{1}{2}$

Fig. 5. Global region and determination of slopes.

The optimal path layout defined a measure of dissimilarity between the two sequences of features, whose general form is

$$D(a,b)=\min_{F}\left[\frac{\sum_{k=1}^{K} d(c(k))\cdot w(k)}{\sum_{k=1}^{K} w(k)}\right], \tag{26}$$

where $d(c(k))$ and $w(k)$ are the local distance between the windows $i(k)$ of the reference vector and $j(k)$ of the recognize vector, and a weighting function in k to maintain a flexible way and improve alignment, respectively. The simplified computational algorithm for calculating the distance of DTW is shown below [1] :

```
$D(0,0)=0$
for $i=1:I$
    $D(i,0)=D(i-1,0)+1$
end
for j = 1: J
    $D(0,j)=D(0,j-1)+1$
end
for $i=1:I$
    for $j=1:J$
        $c1=D(i-1,j-1)+d(i,j\,|\,i-1,j-1)$
        $c2=D(i-1,j)+1$
        $c3=D(i,j-1)+1$
        $D(i,j)=\min(c1,c2,c3)$
    end
end
$D(a,b)=D(I,J)$
```

4. Mathematical foundations

4.1 Fisher's Linear Discriminant Ratio FLDR

Fisher's Linear Discriminant Ratio, is used as an optimization criterion in several research fields, including speech recognition, handwriting recognition, and others [1]. Consider the following definitions:

Within-class scatter matrix.

$$S_w = \sum_{i=1}^{M} P_i S_i \tag{27}$$

where S_i is the covariance matrix for class ω_i, and P_i the a priori probability of class ω_i. Trace$\{S_w\}$ is a measure of the average variance of the features, or descriptive elements of the class.

Between-class scatter matrix

$$S_b = \sum_{i=1}^{M} P_i (\mu_i - \mu_0)(\mu_i - \mu_0)^T \tag{28}$$

where μ_0 is the global mean vector

$$\mu_0 = \sum_{i}^{M} P_i \mu_i \tag{29}$$

Trace$\{S_b\}$ is a measure of the average distance of the mean of each individual class from the respective global value.

Mixture scatter matrix

$$S_m = E\left[(\mu_i - \mu_0)(\mu_i - \mu_0)^T\right] \tag{30}$$

S_m is the covariance matrix of the feature vector with respect to the global mean, and E[.] is the mathematical operator of the expected mean value. Based on the just given definitions, the following criteria can be expressed:

$$\begin{gathered} J_1 = \frac{trace\{S_m\}}{trace\{S_w\}} \\ J_2 = \frac{|S_b|}{|S_w|} = \left|S_w{}^{-1} S_b\right| \end{gathered} \tag{31}$$

It can be shown that J_1 and J_2 take large values when the samples in the *l*-dimensional are well clustered around their mean, within each class, and the clusters of different classes are well separated. Criteria in equation (31) can be used to guide an optimization process, since they measure the goodness of data clustered; the data to be clustered could be the set of features representative of the items of a class. Trace$\{S_b\}$ is a measure of the average distance of the mean of each individual class from the respective global value.

Figure 6 shows an example in which the FLDR is evaluated using equation (31); FLDR and the respective values are displayed. Notice that the more the blue clusters are separated from the red cluster, the bigger FLDR value is.

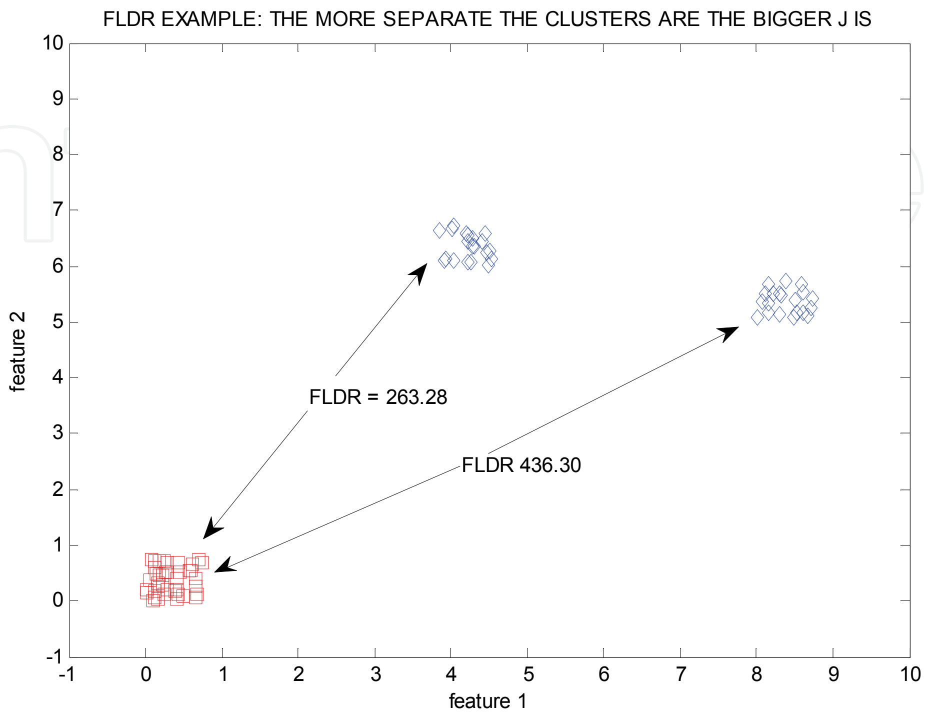


Fig. 6. Example of the FLDR values for two clusters.

4.2 Parseval’s theorem and the Fourier Transform

Parseval’s theorem states, in an elegant manner, that the energy of a discrete signal in the time domain can be calculated in the frequency domain by a simple relation [2], [3]:

$$\sum_{i=0}^{N-1} x[i]^2 = \frac{2}{N}\sum_{k=0}^{N/2} |X[k]|^2 \tag{32}$$

where

N is the number of samples of the discrete signal,
x[i] is the *i*-th sample of the discrete signal,
X[k] is the *k*-th sample of the Fourier transform of x[i].

For a discrete signal x[i], the Fourier transform can be computed using the well known *Discrete* Fourier Transform via the efficient algorithm for its implementation, the FFT [2], [3]:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-2i\pi k\frac{n}{N}} \quad k=0, 1,\ldots,N-1 \tag{33}$$

The implication of the Parseval’s theorem is that an algorithm can search for specific energetic properties of a signal in the frequency domain off-line, and then use the information obtained off-line to configure a bank of digital filters to look for the same energetic properties in the time domain on-line, in real time. The *link* between both domains is the energetic content of the signal.

5. Algorithmic foundations

This section is devoted to describe two important figures in pattern recognition: *backpropagation neural networks* BPNN and *genetic algorithms* GA. The BPNN is used as a reference classifier to compare the performance of the approach presented here to the word recognition problem. The GA is an integral part of the generation of the features in the proposed technique.

5.1 Learning paradigms

There are several major paradigms, or approaches, to machine learning. These include supervised, unsupervised, and reinforcement learning. In addition, many researchers and application developers combine two o more of these learning approaches into one system [23].

Supervised learning is the most common form of learning and is sometimes called programming by example. The learning system is trained by showing it examples of the problem state or attributes along with the desired output or action. The learning system make a prediction based on the inputs and if the output differs from the desired output, then the system is adjusted or adapted to produce the correct output. This process is repeated over and over until the system learns to make accurate classifications or predictions. Historical data from databases, sensor logs, or trace logs is often used as the training or example data.

Unsupervised learning is used when the learning system needs to recognize similarities between inputs or to identify features in the input data. The data is presented to the system, and it adapts so that it partitions the data into groups. The clustering or segmenting process continues until the system places the same data into the same group on successive passes over the data. An unsupervised learning algorithm performs a type of feature detection where important common attributes in the data are extracted.

Reinforcement learning is a type of supervised learning used when explicit input/output pairs of training data are not available. It can be used in cases where there is a sequence of inputs and the desired output is only known after the specific sequence occurs. This process of identifying the relationship between a series of input values and a later output value is called temporal credit assignment. Because we provide less specific error information, reinforcement learning usually takes longer than supervised learning and is less efficient. However, in many situations, having exact prior information about the desired outcome is not possible. In many ways, reinforcement learning is the most realistic form of learning.

Another important distinction in learning systems is whether the learning is done on-line or off-line. On-line learning means that the system is sent out to perform its tasks and that it can learn or adapt after each transaction is processed. On-line learning is like on the job training and places severe requirements on the learning algorithms. It must be very fast and very stable. Off-line learning, on the other hand, is more like a business seminar. You take your salespeople off the floor and place them in an environment where they can focus on improving their skills without distractions. After a suitable training period, they are sent out to apply their new found knowledge and skills. In an intelligent system context, this means that we would gather data from situations that the systems have experienced. We could then augment this data with information about the desired system response to build a training data set. Once we have this database we can use it to modify the behavior of our system.

5.2 Backpropagation Neural Networks

Backpropagation is the most popular neural network architecture for *supervised learning*. It features a *feed-forward* connection topology, meaning that data flow through the network in a single direction, and uses a technique called the *backward propagation* of errors to adjust the connection weights Rumelhart, Hinton, and Williams 1986 in [23]. In addition to a layer of input and output units, a back-propagation network can have one or more layers of hidden units, which receive inputs only from other units, and not from the external environment. A backpropagation network with a single hidden layer or processing units can learn to model any continuous function when given enough units in the hidden layer. The primary applications of backpropagation networks are for prediction and classification.

Figure 7 shows the diagram of a backpropagation neural network and illustrates the three major steps in the training process.

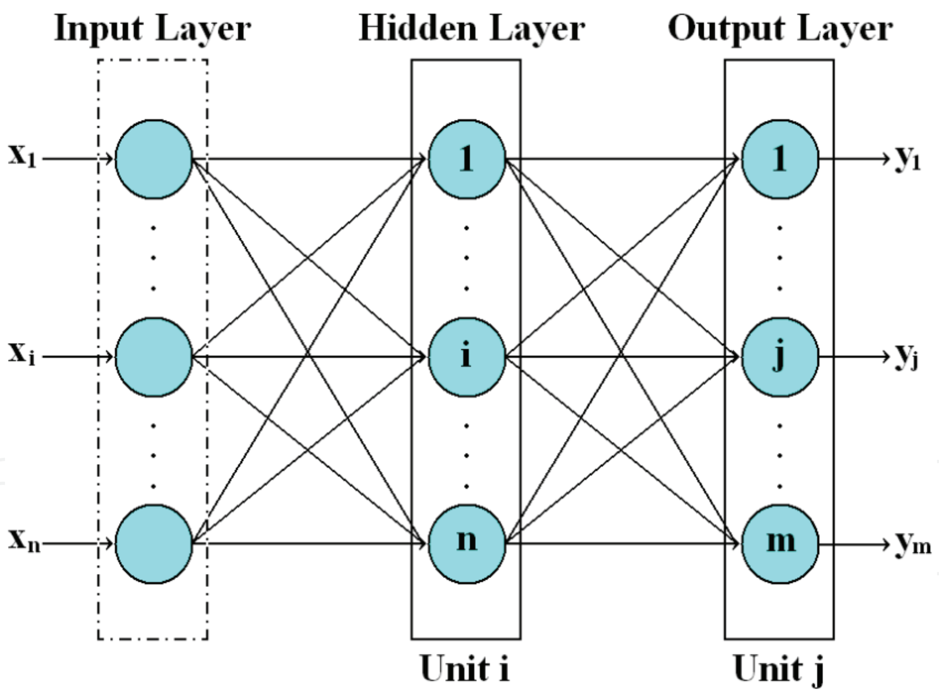


Fig. 7. Topology of a backpropagation neural network.

First, input data is presented to the units of the input layer on the left, and it flows through the network until it reaches the network output units on the right. This is called the forward pass.

Second, the activations or values of the output units represent the actual or predicted output of the network, because this is supervised learning.

Third, the difference between the desired and the actual output is computed, producing the network error. This error term is then passed backwards through the network to adjust the connection weights.

Each network input unit takes a single numeric value, x_i, which is usually scaled or normalized to a value between 0.0 and 1.0. This value becomes the input unit activation. Next, we need to propagate the data forward, through the neural network. For each unit in the hidden layer, we compute the sum of the products of the input unit activations and the weights connecting those input layer units to the hidden layer. This sum is the inner product (also called the dot or scalar product) of the input vector and the weights in the hidden unit. Once this sum is computed, we add a threshold value and then pass this sum through a nonlinear activation function, f, producing the unit activation y_i. The formula for computing the activation of any unit in a hidden or output layer in the network is

$$y_i = f\left(sum_j = \sum x_i w_{ij} + \theta_i\right) \tag{34}$$

where *i* ranges over all the units leading into the *j*-th unit, and the activation function is

$$f\left(sum_j\right) = \frac{1}{1+e^{-sum_j}} \tag{35}$$

As mentioned earlier, we use the *S*-shape sigmoid or logistic function for *f*. The formula for calculating the changes of the weights is

$$\Delta w_{ij} = \eta \delta_j y_i \tag{36}$$

where w_{ij} is the weight connecting unit i to unit j, η is the learn rate parameter, δ_j is the error signal for that unit, and y_i is the output or activation value of unit i. For units in the output layer, the error signal is the difference between the target output t_j and the actual output y_i multiplied by the derivative of the logistic activation function.

$$\delta_j = \left(t_j - y_j\right) f_j^{\prime}\left(sum_j\right) = \left(t_j - y_j\right) y_j \left(1 - y_j\right) \tag{37}$$

For each unit in the hidden layer, the error signal is the derivative of the activation function multiplied by the sum of the products of the outgoing connection weights and their corresponding error signals. So for the hidden unit j.

$$\delta_j = f_j^{\prime}\left(sum_j\right) \sum \delta_k w_{jk} \tag{38}$$

where k ranges over the indices of the units receiving *j*-th unit´s output signal.

A common modification of the weight update rule is the use of a momentum term α, to cut down on oscillation of the weight change becomes a combination of the current weight change, computed as before, plus some fraction (α ranges from 0 to 1) of the previous weight change. This complicates the implementation because we now have to store the weight changes from the prior step.

$$\Delta w_{ij}(n+1) = \eta \delta_j y_i + \alpha \Delta w_{ij}(n) \tag{39}$$

The mathematical basis for backward propagation is described in detail in [23]. When the weight changes are summed up (or batched) over an entire presentation of the training set, the error minimization function performed is called gradient descent.

5.3 Genetic Algorithms

In this section a brief description of a simple genetic algorithm is given. Genetic algorithms are based on concepts and methods observed in nature for the evolution of the species. Genetic algorithms were brought to the artificial intelligence arena by Goldberg [6], [24]. They apply certain operators to a population of solutions of the problem to be solved, in a such a way that the new population is improved compared to the previous one according to a certain criterion function *J* [5], [1], [6], [24]. Repetition of this procedure for a preselected number of iterations will produce a last generation whose best solution is the optimal solution to the problem.

The solutions of the problem to be solved are coded in the *chromosome* and the following operations are applied to the coded versions of the solutions, in this order:

Reproduction. Ensures that, in probability, the better a solution in the current population is, the more replicates it has in the next population,

Crossover. Selects pair of solutions randomly, splits them in a random position, and exchanges their second parts.

Mutation. Selects randomly an element of a solution and alters it with some probability. It helps to move away from local minima.

Besides the coding of the solutions, some parameters must be set up:

N, number of solutions in a population. Fixed or varied.
p, probability with which two solutions are selected for crossover.
m, probability with which an element of a solution is mutated.

The performance of the GA depends greatly on these parameters, as well as on the coding of the solutions in the chromosome. The solutions can be coded in some of the following formats:

Binary. Bit strings represent the solution(s) of the problem. For instance, a chromosome could represent a series of integer indexes to address a database, or the value of a variable(s) that must be integer, or each bit could represent the state (present-absent) of a part of an architecture that is being optimized, and so on.

Real valued. The bit strings represent the value of a real valued variable, in fixed of floating point.

The aspect of one chromosome could be like this: C = {100101010101010101}; the interpretation will vary in accordance with the coding scheme selected to represent the knowledge domain of the problem. For instance, it might represent a set of six indices of three bits each one; or it could have a meaning with all the bits together, representing an 18 bit code.

The primary reason of the success of genetic algorithms is its wide applicability, easy use and global perspective [6], [24], [25]. The next is the listing of a simple genetic algorithm.

```
Procedure (Genetic_Algorithm)
    M = Population size. (*# Of possible solutions at any instance.*)
    $N_g$ = Number of generations. (*# Of iterations.*)
    $N_o$ = Number of offsprings. (*# To be generated by crossover.*)
    $P_\mu$ = Mutation probability. (*# Also called mutation rate $M_r$.*)
    P $\leftarrow \Xi$ (M)
    For j = 1 to M
        Evaluate $f$(p[i])
    EndFor
    For i = 1 to $N_g$
        For (j=1 to $N_o$)
            (x,y) $\leftarrow \emptyset$(p) (*Select two parents x and andfrom current population*)
            Offspring[j] $\leftarrow$ X(x,y)  (*Generate offsprings by crossover of parents x and y*)
            Evaluate $f$(offspring[j]) (*Evaluate  fitness of each offsprings*)
        EndFor
      For j = 1 to $N_0$  (*With probability $p_\mu$ apply mutation*)
          Mutated[j] $\leftarrow \mu$(y)
          Evaluate $f$(mutated[j])
      EndFor
      p $\leftarrow$ Selected(p, iffsprings) (*Select best M solutions from parents & offsprings. *)
    EndFor
    Return highest scoring configuration in p.
End
```

The genetic algorithms find application in the field of speech processing via the solution to the problem of variable and feature selection [11], [14], [26], [27], [28], [29], [30].

6. Restricted-vocabulary speech recognition system

The expression restricted-vocabulary speech recognition refers to the recognition of repetitions of spoken words that belongs to a limited set of words within a semantic field. This means that the words have connected meanings, for instance, the digits = {0,1,2,…, 9} or the days of the week={Saturday, Monday,…, Friday}. The applications of the recognition of limited size word-sets include voice-commanded systems, spoken entry and search for computer databases in warehouse systems, voice-assisted telephone dialing, man-machine interfaces, and others. The advantage of a system developed for a specific semantic field is that it can be built to be much more accurate than those constructed for the general speech recognition, also requiring less extensive training sets.

Restricted-vocabulary speech recognition is also an important research topic because of the intricacies involved in the underlying pattern recognition problem: variable selection, feature generation/selection and classifier selection. Variable selection is mostly restricted to select the raw digitized voice signal as the variable; alternatively, the surrounding environmental noise could be used as another variable for noise cancelation purposes. Feature generation has been carried out by obtaining linear predictive coefficients (LPC), AR/ARMA coefficients, Fourier coefficients, Cepstral coefficients, Mel Spectral Coefficients, and others [31]. In many the cases, the coefficients are computed over a short-time window (typically 10 ms) and over the voiced segments of the speech signal. As well as for the feature generation, for the classifier several options are available: Hidden Markov Models HHM (perhaps the most popular), neural networks (backpropagation, self-organizing maps, radial basis, etc.), support vector and other kernel machines, Gaussian mixtures, Bayesian type classifiers, LVQ, and others. It should be noted that this list of choices of each element in the pattern recognition chain is by no means extensive. Please note that by simply considering all the combinations of variables, features and classifiers mentioned here, it is easily seen that there exist too many ways to implement a restricted-vocabulary speech recognition system.

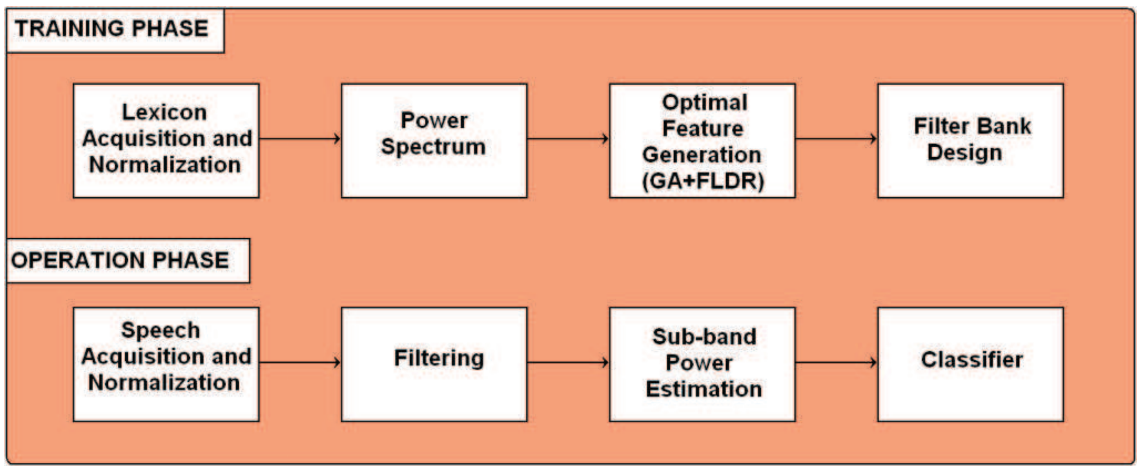

TRAINING PHASE
Lexicon Acquisition and Normalization
Power Spectrum
Optimal Feature Generation (GA+FLDR)
Filter Bank Design
OPERATION PHASE
Speech Acquisition and Normalization
Filtering
Sub-band Power Estimation
Classifier

Fig. 8. Block diagram showing the optimal feature generation with the GA and FLDR.

6.1 Methodology

Figure 8 shows a block diagram of the whole speech recognition system. The system works in two phases: training and operation. The blocks of the training-phase are described below.

Lexicon Acquisition and Normalization. In this stage, the set $\mathbf{L} = \{\mathbf{w_0},\mathbf{w_1},\mathbf{w_2},\ldots, \mathbf{w_{M-1}}\}$ of $\mathbf{M}$ words that will comprise the lexicon to be recognized is acquired from the speaker(s) that will use the system; vectors containing the digitized versions of the voice signals will be at the disposal of the next stage.

Power Spectrum. The power spectrum has been traditionally the source of features for speech recognition; here, the power spectrum of the voice signals will be used by the genetic algorithm to determinate discriminant frequency bands.

Optimal Feature Generation. The features selected here are a) the energy $\mathbf{E}$ of eight to twelve frequency regions (sub-bands) of the spectrum, b) the bandwidth (BW) of each sub-band, and c) the central frequency (F_C) of the sub-bands. See Figure 9. Sub-band processing in speech have been previously used, but in different manners [32], [33], [34]. A genetic algorithm with elitism is used here to select each bandwidth (BW) its central frequency (F_c) and a number of sub-bands. The main parameters of the genetic algorithm are listed in table 1. The cost function of the GA is an expression aimed at maximizing the Fisher's linear discriminant ratio (FLDR). The use of the FLDR in the cost function ensures

Parameter	*Value*
Population	20-120
Mutation	Gaussian (1-2 , 1-2) (scale,shrink)
Selection	Elitism
Crossover	Scattered, one-point and two-points

Table 1. Parameters of the Genetic Algorithm.

increasing both, class separability and cluster compactness between classes ω_0 and ω_1, being ω_0 the class of the word to be recognized and ω_1 the class of the rest of the M-1 words. The FLDR was described here in section 4.1, and the expression adopted here is the equation (31):

$$J_2 = \frac{|S_b|}{|S_w|} = |S_w^{-1} S_b|$$

where $|S_b|$ is the determinant of the inter-class covariance matrix and $|S_b|$ is the determinant of the intra-class covariance class. At the end of the evolutionary process the genetic algorithm produced a set of vectors with the parameters [$\boldsymbol{E}$, $\boldsymbol{BW}$, $\boldsymbol{F_c}$] and the number of sub-bands, between 8 and 12.

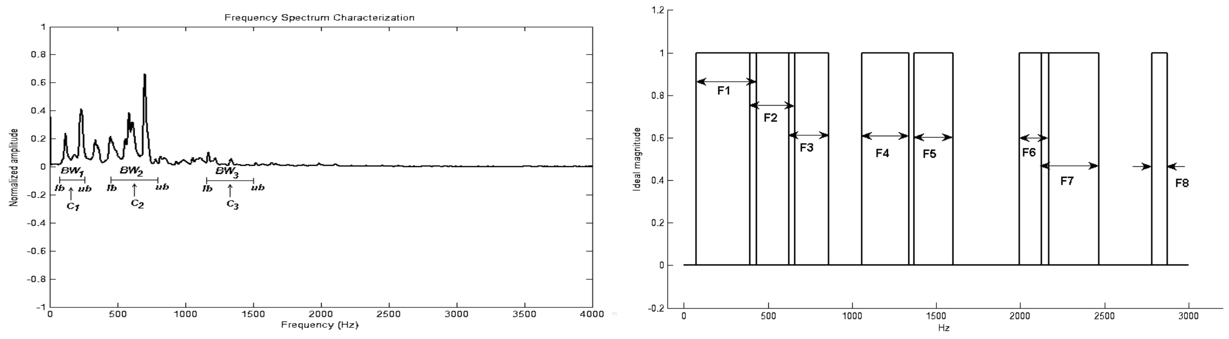


Fig. 9. Sub-band spectrum division to separate the power spectra in discriminant regions.

Bank-filter Design. During the operation phase, the calculation of the power espectrum of the incoming voice signals is not practical for real-time response because the discrete Fourier transform (DFT [2], [3]) requires too much time to be calculated, and the fast Fourier Transform (FFT [2], [3]) requires that the number of samples to be a power of 2, requiring zero padding most of the times. Instead of using the power spectrum, in this application it is used the Parseval's theorem, described in section 4.2, which states that the energy in time and frequency is equal. In discrete form, we recall equation (32):

$$\sum_{i=0}^{N-1} x[i]^2 = \frac{2}{N} \sum_{k=0}^{N/2} |X[k]|^2$$

where *x[i]* is the time domain signal with and *X[k]* is its modified frequency espectrum, which is found by taking the DFT of the signal and dividing the first and last frequencies by the square rooth of two. Therefore, all that have to be done is to filter the sub-bands out of the time signals and then to calculate the energy in each sub-band. This can be perfectly accomplished by a bank of digital band-pass filters whose parameters match the parameters found by the genetic algorithm, and the advantage is that at the end of the last sample of voice in real-time a word can be immediately recognized.

In the Operation Phase, the three first stages operate simultaneously at each time a sample of the voice is acquired, and occur between two successive sampling intervals; in the first stage the voice signal is acquired and a normalization coefficient N_c is updated with the maximum value of the signal; the block labeled filtering represent the action of the filter bank and the outgoing signal is squared and added sample by sample. At the end of a voiced segment of sound, the third block provides to a classifier with the set of features for word classification.

6.2 System design

In this section the details of the system design will be presented.

6.2.1 Characterization of the frequency spectrum using sub-bands

Consider the Fourier Transform of a signal:

$$S(\omega) = F(s(t)) \tag{40}$$

Now consider the frequency spectrum split in sub-bands, as shown in Figure 9. Please notice in Figure 9 that *S*(w*)* has been normalized to unitary amplitude. For each sub-band, the energy can be calculated as:

$$E_i = k_0 \sum_{lb}^{ub} \left(S(\omega)^2 \right) \tag{41}$$

where E_i, is the energy of the *i*-th sub-band, k_0 is a constant proportional to the bandwidth (BW_i) of the *i*-th sub-band, *lb* and *ub* are the lower and upper bounds of the *i*-th sub-band. The feature vector of the *n-th* utterance of *s(t)* in the frequency domain becomes:

$$Sr_n(\omega) = [E_1 \; E_2 \; E_3 \; E_4 \cdots E_M] \tag{42}$$

In which *M* is the number of sub-bands, and Sr_n is the reduced version of the *n*-th *S*(ω*)*. In order to characterize a word in the vocabulary, *N* samples of *s(t)* must be entered. From the example in Figure 9 and without loss of generality, the set of parameters of the respective filter bank that will operate in the time domain is:

$$BF = [C_1 \, BW_1 \, C_2 \, BW_2 \, C_3 \, BW_3 \cdots C_L \, BW_L] \tag{43}$$

The number of filters to be applied in the time-domain signal in this case is L. For a vocabulary of *K* words (*K* small), the spectra in which the solution must be searched is given by *K* matrices of *N* rows and 2000 columns.

6.2.2 Genetic algorithm set up

6.2.2.1 Coding the chromosome

The chromosome is comprised of the parameters of the filter bank described in equation 43. The number of sub-bands is fixed, and each one of the centers and bandwidths are subject to the genetic algorithm. Real numbers are used.

6.2.2.2 The cost function

The cost function is given by equation (32), criterion *J2*. The goal is to maximize J_2 as a function of the centers and bandwidths.

6.2.2.3 Restrictions

The following restrictions apply:

R1. Sub-bands overlapping < = 50Hz,

R2. Bandwidth is limited to range from 40 to 400 Hz, varying according to the performance of the genetic algorithm.

6.2.2.4 Operating parameters of the genetic algorithm

The main parameters of the genetic algorithm are summarized in the Table 1. The values of the parameters are given according to the best results obtained by experimentation. The genetic algorithm was ran in Matlab®, using the genetic algorithms toolbox and the *gatool* guide. The nomenclature of the parameters in Table 1 is the one used by Matlab®.

6.2.2.5 Application's algorithm

To make operational de methodology described so far, the following steps apply:

1. *Vocabulary definition.* 2 to K words. In many real life applications, K in 8 to 15 words do the job.
2. *Database acquisition.* 15 to 20 utterances of each word from the vocabulary, for learning purposes. The sampling frequency can be set from 6000 to 8000 Hz. Human voice accommodates easily here.
3. *s(t)* to *S*(w) transformation. Apply Fourier transform to the data, normalize to unitary amplitude and to a fixed length of 2000.
4. *Data preparation for the GA.* Set-up the size (eq. 43) and restrictions (subsection 6.2.2.3) of the filter bank (chromosome).
5. *Running the GA.* Run the GA to find the sub-bands whose J_2 (eq. (32)) is the maximum.
6. *Filters realization.* For each sub-band, compute the coefficients of the respective bandpass filters. For real-time implementation, order from 4 to 8 is recommended, type IIR, elliptic. Elliptic filters achieve great discrimination and selectivity. Implementation details can be found in [2].
7. *Modeling the commands.* To make comparisons and therefore classification, a Gaussian statistical model of each word is to be constructed for each command in the vocabulary. Proceed as follows for each command:
   a. Construct a matrix *C* of 15-20 rows of Sr_n(w) and *M* columns (one row per sample of the command, one column per sub-band selected by the AG),
   b. Compute the mean value overall the samples, to find the average energy per sub-band, this is the feature vector of the command (μ_i)

c. Compute the covariance matrix, $S_i = \mathrm{cov}(C_i)$.
d. At any time, the Mahalanobis distance between the model of the *i*-th command and the feature vector ***x*** of an incoming command is:

$$dM(x,\mu_i) = (x-\mu_i)^*\Sigma_i^{-1*}(x-\mu_i)^{-1} \tag{44}$$

To make the real-time implementation, a digital system must sample the input microphone continuously. Each sample of a command must be filtered by each filter (sub-band extraction). The integral of the energy of the signal leaving each filter over the period of the command is used to create the feature vector of that command, that is, *x* in equation 44. The feature vector is compared to each model in the vocabulary, and the command is recognized as the one with the minimum *dM* score. This is the so-called "minimum distance classifier" [5].

7. Results

To test the system, the following lexicons were used: L_0={faster, slower, left, right, stop, forward, reverse, brake}, L_1={zero, one, two, three, four, five, six, seven, eigth, nine}, L_2={rápido, lento, izquierda, derecha, alto, adelante, reversa, freno}, L_3 = {uno, dos, tres, cuatro, cinco}. In all the lexicons, 3 male and 3 female volunteers were enrolled. They donated 116 samples of each word, 16 for training and 100 for testing. To demostrate the power of our approach we used the minimum distance classifier with the Mahalanobis distance. In all the cases, the genetic algorithm was ran 30 times to find the best response in the training set. During the training phase, the leave-one-out method was used to exploit the limited size of the training set [1]. Table 2 summarizes the results. In columns 5 and 7 are shown the comparison against a backpropagation neural network using as features Cepstral coefficients. The experiments were done using Matlab(R) and its associated toolboxes of genetic algorithms, neural networks and digital signal processing. The real-time implementation was done with a TMS320LF2407 Texas Instruments(R) Digital Signal Processor mounted on an experimentation card.

			Simulations		Real-time on DSP	
			MDC[3]	BPNN[4]	MDC[3]	BPNN[4]
G[1]	**L[2]**	**Training Set**	**Testing Set**	**Testing Set**	**Real scenario**	**Real scenario**
Female	0	100	97	92	94	90
	1	100	98	90	95	90
	2	100	100	97	94	89
	3	100	100	91	95	88
Male	0	100	100	94	96	89
	1	100	100	90	95	90
	2	100	99	92	94	90
	3	100	98	92	93	88

[1]Gender, [2] Lexicon, [3]Minimum distance classifier, [4]Backpropagation neural network [1] 8-32-K neurons per layer, K according to the experiment, one neuron for each word in L.

Table 2. Percentage (%) of correct classification with 4 lexicons, 2 languages, 6 persons, male and female voices, 2 classifiers. Simulations and real time implementation.

7.1 Results and discussion

7.1.1 Results in the L_3 Spanish vocabulary

The genetic algorithm was executed 30 times, and the maximum Fisher's ratio obtained was 62. The resulting best chromosome was:

$$BF = [254\ 180\ 526\ 132\ 744\ 118\ 1196\ 141\ 1483\ 115\ 2082\ 86\ 2295\ 171\ 2828\ 46]$$

From which the corresponding center and bandwidth were:

$$\begin{aligned} C &= [254\ 526\ 744\ 1196\ 1483\ 2082\ 2295\ 2828] \\ BW &= [180\ 132\ 118\ 141\ 115\ 86\ 171\ 46] \end{aligned}$$

The recognition rates in the training and testing sets were 100% and 99%, respectively. In real conditions the correct classification rate was 93.5% in 40 repetitions of each word to the microphone.

7.1.2 The genetic algorithm in L_3

Consecutive executions of the GA produced variable Fisher's ratios. It was observed here that the population size is critical, since a population of 20 chromosomes produced Fisher's ratios between 35 and 42, while a population of 120 individuals easily produced Fisher's ratios above 58. It was noticed during experimentation that specific values for restrictions R1 and R2 also have a strong influence in the outcome.

Comparing the results over a traditional approach with neural networks and cepstral coefficients it is evident a higher performance and, more important, the system exhibits real-time operation and very low computational effort compared to neural networks and real-time computation of the Cepstral coefficients.

7.1.3 Results in the L_0 English vocabulary

The genetic algorithm was executed 30 times, varying population size, probability of mutation, restrictions, number of sub-bands, and other parameters. The initial number of sub-bands was 8, then it was scaled to 9 and 12; in this scenario, the Fisher's ratio varied from 23 (8 sub-bands, population size = 20) to 48 (12 sub-bands, population size = 200). The resulting centers and bandwidths were:

$$\begin{aligned} C &= [250\ 446\ 648\ 1283\ 1483\ 1776\ 2018\ 2506\ 2737\ 3197\ 3383\ 3833] \\ BW &= [\ 5\ 138\ 187\ 157\ 139\ 43\ 207\ 106\ 105\ 98\ 148\ 224] \end{aligned}$$

Intra-class repeatability and inter-class differences

The performance in the training set was 100%; the performance in the testing set was 99%. The correct classification per word was {100% 100% 98% 100% 97% 100% 98% 99%} for the respective words {*'faster','slower','left','right','stop','forward','reverse','brake'*}, respectively. Figure 10 shows the normalized espectra of two utterances of the word 'faster' and the word 'slower'. In both cases notice the repeatability in the frequency domain, as well as the difference between both sets of spectra.

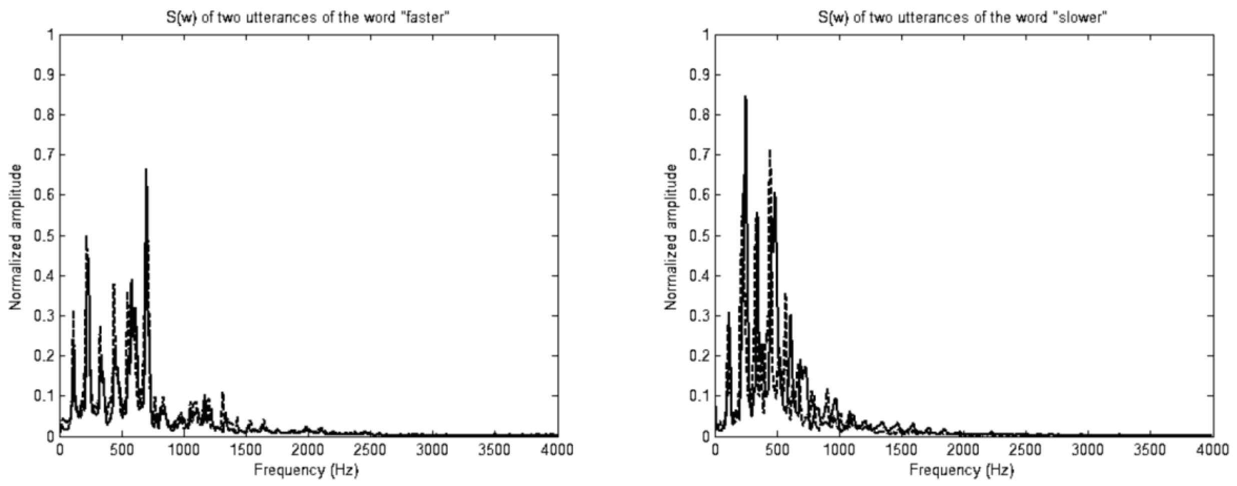


Fig. 10. Normalized spectra of the words “faster” and “slower”.

7.1.4 Results of the real-time implementation

A minimum distance classifier was implemented in a digital signal processor TMS320LF2407 for each of the four lexicons **L**={L_0, L_1, L_2, L_3} from Texas Instruments, in order to verify the performance using in a nearly real-life application. The voiced/no-voiced segmentation was performed using a push-button to start and finish capturing the voice. The DSP has a built-in 10-bit analog to digital converter facilitating the interfacing task. The digital filters used were IIR topology, elliptic type, 8th order. The filter coefficients (*A, B*) were calculate using the Matlab® Software. The analog-to-digital conversion was set-up to acquire one sample every T seconds, (T=1/6000); each time a sample came into de device, the filters actuated and the respective output was squared and accumulated to calculate the energy of the signal. Scaling issues had to be solved since the model was created in a real valued [-1 , 1] scale, while the DSP just “see“ integer values. Once a whole command was processed, it was just a matter of a few miliseconds to apply the minimum distance classifier and provide the classification. The correct classification rate was in this case of the order of 94.5%, in a total of 1200 repetitions of the words in **L**.

8. Conclusions and future work

In this chapter was presented a method to implement a high performance, real-time, restricted-vocabulary speech recognition system, combining a genetic algorithm and the Fisher’s Linear Discriminant Ratio (FLDR) in its matrix formulation. A review of the concepts of variable and feature selection as well as feature generation was made; also were presented some concepts related with speech processing, like the LPC formulation and the DTW method for template matching.

One of the conceptual tools used here was the energy of the signal in certain sub-bands in the frequency domain; thanks to the Parseval’s theorem, the same amounts of energy can be calculated in the time domain via a bank of digital filters, enabling thus a very fast way to apply the recognizer, since the process goes on at the same time as the occurrence of the word is exerted. Mainly, two experiments were shown, in Spanish and English, with male and female participants; in both cases high performance was attained, beyond 94% at the worst case. Compared to a typical implementation with backpropagation neural networks and cepstral coefficients, this approach was at least 10% more effective in near real-life application.

The genetic algorithm consistently maximized the criteria of inter-class separability and intra-class compactness, under different conditions of population, probability of mutation, etc., and also varying the restriction set. It is remarkable and worth to mention that the genetic algorithm didn't gave the best result in its first execution, which means that the execution must be repeated to achieve good results; increasing the population and manipulating the restriction set demonstrated that it is possible to obtain a variety of different outcomes, so it is important to experiment carefully.

The future work will consist of developing the voice/unvoiced detection in noisy environments, investigate an adapt more features that can be easily computed in time and with a dual in frequency, start working on the non-dependant speaker approach, making use of more robust classifiers, and finally, increase the vocabulary size, although still restricted to a specific semantic field, like in [35]. Another interesting venue is the one in which the user aging process is taken into account by the speech recognition system [36].

9. Acknowledgment

The authors wish to express their gratitude for financial support of this project to *Universidad Politécnica de Aguascalientes*.

10. References

[1] K. Koutroumbas and S. Theodoridis, *Pattern Recognition*, 1st ed. California, E. U. A.: Academic Press, 1999.

[2] John G. Proakis and Dimitris G. Manolakis, *Digital Signal Processing. Principles, Algorithms, and Applications.*, 3rd ed. New Jersey, U.S.A.: Prentice Hall, 1996.

[3] L. R. Rabiner and R. W. Schafer, *Introduction to Digital Signal Processing*, 1st ed. Hannover, U.S.A.: Now Publishers Inc., 2007.

[4] Isabelle Guyon and André Elisseeff, An Introduction to Variable and Fature Selection, *Journal of Machine Learning Research*, vol. 3, pp. 1157-1182, 2003.

[5] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 1st ed. New York, U.S.A.: John Wiley & Sons, Inc., 2001.

[6] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. U. S. A.: Addison-Wesley Professional, 1989.

[7] F. Itakura, Minimum Prediction Residual Principle applied to Speech Recognition, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, no. 2, pp. 67-72, 1975.

[8] J. Makhoul, Linear Prediction: A Tutorial Review, *Proceedings of IEEE*, vol. 63, no. 4, pp. 561-580, 1975.

[9] C. D. Manning, *Foundations of Statistical Natural Language Processing*, 6th ed. Cambridge, Massachussets, U.S.A.: MIT Press, 2003.

[10] J. Ramírez, J. M. Górriz, and J. C. Segura, Voice Activity Detection. Fundamentals and Speech Recognition System Robustness, in *Robust Speech Recognitin and Understanding*, M. Grimm and K. Kroschel, Eds. Vienna, Austria: In-Tech, 2007, cap. 1, pp. 1-22.

[11] L. D. Vignolo, H. L. Rufiner, D. H. Milone, and J. C. Goddard, Evolutionary Splines for Cepstral Filterbank Optimization in Phoneme Classsification, in *EURASIP Journal on Advances in Signal Processing*, vol. 2011, 2011, pp. 1-15.

[12] T. Takiguchi, N. Miyake, H. Matsuda, and Y. Ariki, Voice and Noise Detection with AdaBoost, in *Robust Speech Recognition and Understanding*, M. Grimm and K. Kroschel, Eds. Vienna, Austria: In-Tech, 2007, cap. 4, pp. 67-74.

[13] S. Y. Suk and H. Kojima, Voice Activated Appliances for Severely Disabled Persons, in *Speech Recognition, Technologies and Applications*, F. Mihelic and J. Zibert, Eds. Vienna, Austria: In-Tech, 2008, cap. 29, pp. 527-538.

[14] R. Cardin, Improved Learning Strategies for Small Vocabulary Automatic Speech Recognition, McGill University, Montreal, Quebec, Canadá, Doctor of Philosophy Thesis 1993.

[15] Isabelle Guyon. (2011, june) Isabelle Guyon's home page. [on-line]. Hyperlink http://www.clopinet.com/ isabelle/.

[16] I. S. Oh, J. S. Lee, and B. R. Moon, Hybrid Genetic Algorithms for Feature Selection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1424-1437, November 2004.

[17] P. R. Somol, J. Novovicova, and P. Pudil, Efficient Feature Subset Selection and Subset Size Optimization, in *Pattern Recognition Recent Advances*, A. Herout, Ed. Vienna, Austria: InTech, 2010, cap. 4, pp. 75-98.

[18] Y. Sun, S. Todorovic, and S. Goodison, Local-Learning-Based Feature Selection for High-Dimensional Data Analysis, *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1610-1626, September 2010.

[19] J. Teixeria de Souza, R. A. Ferreira do Carmo, and G. Campos de Lima, On the Combination of Feature Selection and Instance Selection, in *Machine Learning*, Y. Zhang, Ed. Vienna, Austria: In-Tech, 2010, cap. 9, pp. 158-171.

[20] X. Huang, A. Acero, and H. W. Hon, *Spoken Language Processing. A guide to Theory, Algorithm, and System Development*, 1st ed. New Jersey, U.S.A.: Prentice Hall PTR, 2001.

[21] A. Zacknich, *Principles of Adaptive Filters and Self-Learning Systems*, 1st ed. London, England: Springer, 2005.

[22] H. Sakoe and S. Chiba, Dynamic Programming Algorithm Optimization for Spoken Word Recognition , *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43-49, 1978.

[23] P. J. Bigus and J. Bigus, *Constructing Intelligent Agents with Java. A Programmer´s Guide to Smarter Applications.*, 1st ed.: John Wiley & Sons, Inc., 2001.

[24] K. S. Tang, K. F. Man, S. Kwong, and Q. He, Genetic Algorithms and their Applications, *IEEE Signal Processing Magazine*, pp. 22-37, November 1996.

[25] S. M. Sait and A. Youssef, *Iterative Computer Algorithms with Applications in Engineering.*, 1st ed. Los Alamitos, Cal., U.S.A.: IEEE Computer Society, 1999.

[26] S. M. Ahadi, H. Sheikhzadeh, R. L. Brennan, and G. H. Freeman, An Effective Front-End for Automatic Speech Recognition, in *2003 International Conference on Electronics, Circuits and Systems*, Sharah, United Arab Emirates, 2003, pp. 1-4, Sub-band speech recognition.

[27] M. P. G. Saon, G. Zweig, J. Huang, B. Kingsbury, and L. Mangu, Evolution of the Performance of Automatic Speech Recogntion Algorithms in Transcribing

Conversational Telephone Speech, in *IEEE Instrumentation and Measurement Technology Conference*, Budapest, Hungary, 2001, pp. 1926-1931.

[28] S. Kwong, Q. H. He, K. F. Man, K. S. Tang, and C. W. Chau, Parallel Genetic-based Hybrid Pattern Matching Algorithm for Isolated Word Recognition, *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 12, no. 4, pp. 573-594, 1998.

[29] V. V. Ngoc, J. Whittington, and J. Devlin, Real-time Hardware Feature Extraction with Embedded Signal Enhancement for Automatic Speech Recognition, in *Speech Technologies*. Vienna, Austria: In-tech, 2011, cap. 2, pp. 29-54.

[30] S. A. Selouani and D. O'Shaughnessy, Robustness of Speech Recognition using Genetic Algorithms and Mel-cepstral Subspace Approach, in *IEEE International Conference on Acoustics, Speech, and Singal Processing 2004 ICASSP '04*, Montreal, Quebec, Canada, 2004, pp. I-201-4.

[31] L. R. Rabiner, Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-286, 1989.

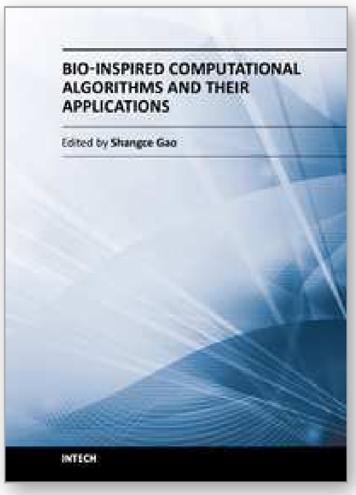
[32] S. Okawa, E. Bocchieri, and A. Potamianos, Multi-Band Speech Recognition in Noisy Environments, in *International Conference on Acoustics, Speech and Signal Processing ICASSP 1998*, Prague, Czech Republic, 1998, pp. 1-4.

[33] A. de la Torre et al., Speech Recognition Under Noise Conditions: Compensation Methods, in *Speech Recognition and Understanding*, M. Grimm and K. Kroschel, Eds. Vienna, Austria: In-Tech, 2007, cap. 25, pp. 440-460.

[34] A. Álvarez et al., Application of Feature Subset Selection Based on Evolutionary Algorithms for Automatic Emotion Recognition in Speech, *Lecture Notes in Computer Science. Advances in Nonlinear Speech Processing,* vol. 4885, pp. 273-281, May 2007.

[35] G. E. Dahl, D. Yu, L. Deng, and A. Acero, Large Vocabulary Continuous Speech Recognition with Context Dependent DBN-HMMS, in *International Conference on Acoustics, Speech and Signal Processing ICASSP 2011*, Prague, Czech Republic, 2011, pp. 1-4.

[36] B. M. Ben-David et al., Effects of Agging and Noise on Real-Time Spoken Word Recognition: Evidence from Eye Movements, *Journal of Speech, Language, and Hearing Research*, vol. 54, pp. 243-262, February 2011.


BIO-INSPIRED COMPUTATIONAL ALGORITHMS AND THEIR APPLICATIONS
Edited by Shangce Gao
INTECH

Bio-Inspired Computational Algorithms and Their Applications
Edited by Dr. Shangce Gao

ISBN 978-953-51-0214-4
Hard cover, 420 pages
Publisher InTech
Published online 07, March, 2012
Published in print edition March, 2012

Bio-inspired computational algorithms are always hot research topics in artificial intelligence communities. Biology is a bewildering source of inspiration for the design of intelligent artifacts that are capable of efficient and autonomous operation in unknown and changing environments. It is difficult to resist the fascination of creating artifacts that display elements of lifelike intelligence, thus needing techniques for control, optimization, prediction, security, design, and so on. Bio-Inspired Computational Algorithms and Their Applications is a compendium that addresses this need. It integrates contrasting techniques of genetic algorithms, artificial immune systems, particle swarm optimization, and hybrid models to solve many real-world problems. The works presented in this book give insights into the creation of innovative improvements over algorithm performance, potential applications on various practical tasks, and combination of different techniques. The book provides a reference to researchers, practitioners, and students in both artificial intelligence and engineering communities, forming a foundation for the development of the field.