Chapter 8
Evolutionary Algorithms and Speech Recognition

Abstract In this chapter, we present an approach for optimizing the front-end processing of ASR systems by using Genetic Algorithms (GAs). The front-end uses a multi-stream approach to incorporate, in addition to MFCCs, auditory-based phonetic distinctive cues. These features are combined in order to limit the impact of the speech signal degradations due to interfering noise. Some of many advantages of using GAs include the possibility to improve the robustness without modifying the recognition system models and without estimating environment parameters, such as the noise variance and/or stream weights. The co-existence, in two streams, of the two types of front-end parameters (MFCCs and distinctive cues) is also managed by the GA. The evaluation is carried out by using a noisy version of the TIMIT corpus.

Keywords Hidden Markov Models • Genetic Algorithms • KLT • Variance of reconstruction error • Acoustic indicative features • TIMIT corpus

8.1 Expected Advantages

Evolutionary computation is a class of soft computing derived from biological concepts and evolution theory. Systems using evolutionary principles model a problem in such a way that the solution is optimized and it keeps improving over time. Evolutionary Algorithms (EAs), that are the most important sub-fields of evolutionary computing, can be considered as heuristic search techniques based on the principles of natural selection. EAs involve various populations of solutions that undergo transformations by using genetic operators that help to converge to the best solution. Selection plays an important role in the evolutionary-based process. In most applications, the determination of the selection function requires an explicit evaluative function which should be interpretable and meaningful in terms of performance, and it is usually prepared by the human designer [45].

Evolutionary Algorithms include genetic algorithms, evolution strategies, evolutionary programming and genetic programming. They have been successfully applied to various domains including machine learning, optimization, bioinformatics

S.-A. Selouani, *Speech Processing and Soft Computing*, SpringerBriefs in Electrical and Computer Engineering, DOI 10.1007/978-1-4419-9685-5_8,

and social systems [37]. However, their use in speech recognition is still very limited probably because it is very difficult to design human evaluation explicit functions for speech recognition systems since we cannot know in advance what utterance will be spoken. Among the EAs, Genetic Algorithms (GAs) have become an increasingly appreciated and well-understood paradigm beyond the soft computing community [127]. In the field of speech recognition robustness, investigating innovative strategies becomes essential to overcome the drawbacks of classical approaches. For this purpose, GAs can constitute robust solutions as they have demonstrated their power to find optimal solutions in complex problems. The main advantage of GAs is their relative simplicity.

Let's consider $P(t)$ as a population of individuals at time t, $\Phi(.)$ as a random operator and $\Psi(.)$ as the individuals' selection function. The procedure underlying the GA which leads to the population of the next generation can be formalized by the following equation:

$$P(t+1) = \Phi(\Psi(P(t))) \tag{8.1}$$

It should be mentioned that in contrast to other formal methods, the performance of GA is not impacted by the representation of the population. Thanks to the $\Psi(.)$ function, GAs also offer the possibility to incorporate prior(human) knowledge of the problem, thus yielding to better solution accuracy. GAs can also be combined to other soft computing and optimization techniques (e.g. tuning neural networks structure [26]) by modifying the $\Phi(.)$ operator.

In contrast to many other optimization methods, parallelization of the GAs is possible. In some applications, parallel implementations are necessary to reach high-quality solutions in a reasonable time span [43]. Another important advantage is the ability of GAs to be robust towards dynamic changes. They also do not require a complete restart of the process when an environment change occurs [3]. In previous work [131], we have demonstrated the efficiency of a solution dealing with genetic optimization of NN-based digit recognizer. To improve the robustness of speech recognition, we investigate the hybridization of GAs with the KLT subspace decomposition using the VRE method (described in Chapter 4). This approach uses local search information and mechanisms to achieve complementarity between the genetic algorithm optimization and KLT-VRE subspace decomposition.

8.2 Problem Statement

The principle of GAs consists of manipulating a population of solutions and implementing a 'survival of the fittest individual' strategy to find the best solution. Simulating generations of populations, the fittest individuals of any population are encouraged to reproduce and survive across successive generations to improve both the overall and individual performance. In some GA implementations a proportion of less performing individuals can survive and also reproduce. A more complete presentation of GAs can be found in [129].

Limiting the drop in speech recognition performance in the context of acoustic environment changes remains one of the most challenging issues of speech recognition in practical applications. This misperformance is due to the unpredictability of adverse conditions that create mismatches between the training data and the test data used by the recognizers. New strategies are needed to make the ASR not only robust but also capable of self-adaptation to variable acoustic conditions. In the ideal situation, the ASR is expected to be capable of perceiving the environment changes and to adapt key features (models, front-end process, voice activity detector parameters,...) to the new context. In their pioneering work, Akbacak and Hansen proposed an original framework called *Environmental Sniffing* that aims to perform smart tracking of environmental conditions and to guide the ASR engine to the best local solution adapted to each environmental condition [7]. In this chapter, we investigate the use of GAs in order to optimize the front-end processing of ASR systems. The front-end features are composed of MFCCs and auditory-based phonetic distinctive cues. The expected advantages of using GAs is that the ASR robustness can be improved without modifying the recognition models and without determining the noise variance.

8.3 Multi-Stream Statistical Framework

HMMs constitute the most successful approach developed for modeling the statistical variations of speech in an ASR system. Each individual phone (or word) is represented by an HMM. In large-vocabulary recognition systems, HMMs usually represent subword units, either context-independent or context-dependent, to limit the amount of training data and storage required for modeling words. Most recognizers typically use left-to-right HMMs, which consist of an arbitrary number of states N. The output distribution associated with each state is dependent on one or more statistically independent streams. To integrate the proposed features in the input vector, we merged different sources of information about the speech signal extracted from both the cepstral analysis and Caelen's auditory-based analysis. The multi-stream paradigm is used for modeling the statistical variations of each information source (stream) in an HMM-based ASR system [135]. In this paradigm, an observation sequence $\mathbf{O}$ composed of S input streams, $\mathbf{O}_s$ possibly of different lengths, is assumed as representative of the utterance to be recognized, and the probability of the composite input vector $\mathbf{O}_t$ at a time t in state j can be written as follows:

$$b_j(\mathbf{O}_t) = \prod_{s=1}^{S} [b_{js}(\mathbf{O}_{st})]^{\gamma_s} \tag{8.2}$$

where $\mathbf{O}_{st}$ is the input observation vector in stream s at time t and γ_s is the stream weight. Each individual stream probability $b_{js}(\mathbf{O}_{st})$ is represented by the most common choice of distribution, *the multivariate mixture Gaussian*:

$$b_{js}(\mathbf{O}_{st}) = \sum_{m=1}^{M} c_{jsm}\, \mathcal{N}(\mathbf{O}_{st}; \mu_{jsm}, \Sigma_{jsm}) \tag{8.3}$$

where M is the number of mixture components in stream s, c_{jsm} is the weight of each mixture component of state j in each mixture of each stream and $\mathcal{N}(\mathbf{O};\mu,\Sigma)$ denotes a multivariate Gaussian of mean μ and covariance Σ and can be written as:

$$\mathcal{N}(\mathbf{O};\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^n|\Sigma|}}\exp^{-\frac{1}{2}(\mathbf{O}-\mu)'\Sigma^{-1}(\mathbf{O}-\mu)} \tag{8.4}$$

In the multi-stream HMM the fusion is assumed to be performed by a single global likelihood probability. The observations are assumed to be generated by each HMM stream with identical topologies and modeled as mixtures of Gaussian densities. In the application presented in this chapter, three streams are considered: MFCCS, the MFCC derivatives and the Caelen Distinctive Cues (CDCs) described in Section 6.3.2.

8.4 Hybrid KLT-VRE-GA-based Front-End Optimization

The principle of the signal subspace techniques consists of constructing an orthonormal set of axes forming a representational basis that projects towards the direction of maximum variability. Applied in the context of noise reduction, these axes permit us to decompose the space of the noisy signal into a signal-plus-noise subspace and a noise subspace. As seen previously, the enhancement is performed by removing the noise subspace and estimating the clean signal from the remaining signal space.

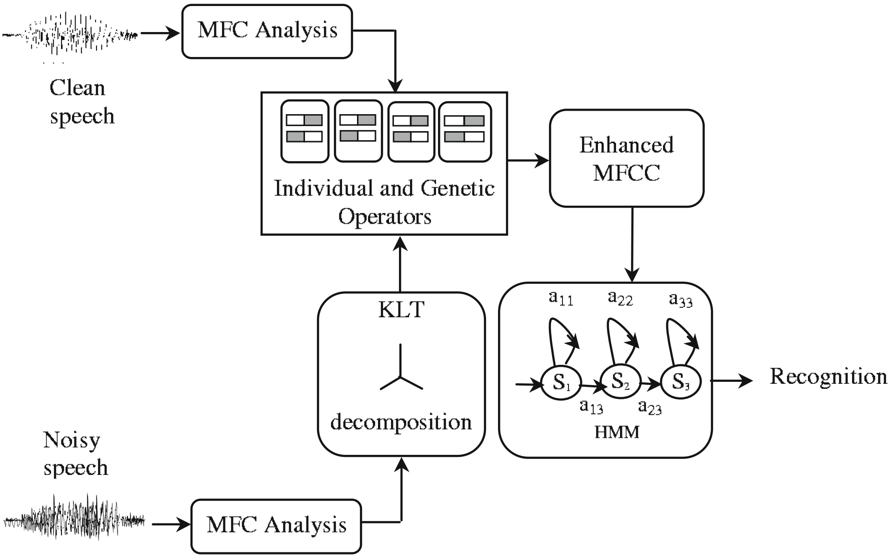

Fig. 8.1 General overview of the hybrid KLT-VRE-GA ASR system.

In Chapter 5, we have described the framework of evolutionary subspace filtering. This framework is extended in this chapter to the ASR robustness by using GAs to optimize the subspace decomposition of the multi-stream vector. By using GAs, no empirical or *a priori* knowledge is needed at the beginning of the evolution process. As illustrated in Figure 8.1, a mapping operator using a Mel-frequency subspace decomposition and GAs is performed. This evolutionary eigendomain transformation attempts to achieve an adaptation of ASR systems by tracking the best KLT reconstruction after removing the noise subspace using the VRE method.

8.5 Evolutionary Subspace Decomposition using Variance of Reconstruction Error

As shown in Chapter 4, the subspace filtering provides the clean speech estimate, $\hat{\mathbf{s}} = \mathbf{Hx}$, where $\mathbf{H} = \mathbf{UGU^T}$ is the enhancement filter containing the weighting factors applied on the eigenvalues of the noisy speech $\mathbf{x}$ corrupted by the noise $\mathbf{n}$. In the evolutionary eigendecomposition, the $\mathbf{H}$ matrix is replaced by $\mathbf{H_{gen}} = \mathbf{UG_{gen}U^T}$. The diagonal matrix $\mathbf{G_{gen}}$ contains the weighting factors optimized by the genetic operators. Therefore, to improve the ASR performance, the task prior to recognition is finding an estimate for $\mathbf{s}$ along the direction ξ_j to best correct the noise effect by using the VRE technique. The number of optimal principal components (PCs) is obtained by achieving the minimum reconstruction error. The reconstruction is performed by using eigenvectors weighted by the optimal factors of the $\mathbf{G_{gen}}$ matrix. These factors will constitute the individuals of a given population in the GA process. The mechanism of determining the optimal PCs is performed on many populations in order to achieve the best reconstruction over a wide range of solutions. As specified in Chapter 4, the variance of the reconstruction error in all directions and dimensions can be calculated by:

$$VRE(l) = \sum_{j=1}^{N} \frac{u_j(l)}{\xi_j^T R \xi_j}. \tag{8.5}$$

The *VRE* is calculated by considering the variances u_j (corresponding to the eigenvalues) in all directions and using the R_{xx} the autocorrelation matrix of the noisy signal. This *VRE* has to be minimized to obtain the best reconstruction and will be included in the GA process as an objective function.

8.5.1 Individuals' Representation and Initialization

Any application based on GAs requires the choice of gene representation to describe each individual in the population. In our case the genes are the components of

$\mathbf{H_{gen}}$ matrix elements. The real-valued representation is suitable and is expected to obtain more consistent results across replications. An alphabet of floating point numbers has values ranging within the upper and lower bounds of $+1.0$ and -1.0 respectively. This representation is closer to the real representation of the weight factors, and will facilitate the interpretation of optimization results.

To start the evolution process, a pool containing a population of individuals representing the weight factors, g_i is created. Ideally, this pool is initialized with a zero-knowledge assumption by using a population of completely random values of weights. Another approach (guided) consists of performing a first KLT subspace decomposition and using the principal components obtained to constitute the first population pool. Regardless to the initialization method (random or guided), these individuals evolve through many generations in the pool where genetic operators are applied. Some of these individuals are selected to survive and to reproduce according to their performance and other considerations that are taken to insure a good coverage of the solution space. The individuals' performance is evaluated through the use of an objective function.

8.5.2 Selection Function

Various methods exist for the selection of individuals to produce successive generations [127]. Most approaches are based on the assignment of a probability of selection, $Prob_j$ to each individual, j, according to its performance. To perform the selection of the weight factors in the pool, we use a new variant of the normalized geometric ranking method originally proposed in [65]. The probability of selection $Prob_j$ is given by:

$$Prob_j = \frac{q(1-q)^{s-1}}{1-(1-q)^{Pop}}, \tag{8.6}$$

where q is the probability of selecting the best individual, s is the rank of the individual (1 is the rank of the best individual), and *Pop* is the population size. All solutions are sorted and ranked. To insure the solution diversity, a mechanism giving more chance of selection to a small proportion of worse performing individuals is applied. This mechanism is formalized as follows:

$$Prob_j^{'} = \begin{cases} Prob_j & \text{for (s} = 1, ..., \text{Pop}/2) \\ Prob_{j+\mathrm{Pop}/2-k} & \text{for (s} = \text{Pop}/2, ..., \text{Pop) and (k} = 0, ..., \text{Pop}/2) \end{cases} \tag{8.7}$$

In this selection method, the pool is divided into two groups. The first group contains individuals that perform better than the elements of the second group. The regular ranking, according to the fitness value, is applied in this first group. In the second group, the ranking is inverted in order to ensure and maintain the population diversity by giving a chance to some of the less performing individuals to be selected. This diversity allows the GA to perform fruitful exploration by expanding the search space of solutions. Therefore, in the selection process, $Prob'_j$ is used to select individuals.

8.5.3 Objective Function

The performance of any individual k is measured by an objective function $\mathcal{F}(k)$ also called the fitness function. In GAs two types of objective functions can be considered. The first type is the best fitness, where the retained solution corresponds to the individual having the best performance. The second type is average fitness, which provides the solution corresponding to the average of the best individuals after a certain number of runs. The objective (fitness) function is defined in terms of a performance measure and gives a quantifiable way to rank the solutions from good to bad. In the KLT-VRE-GA framework, the best solution corresponds to the individual minimizing the *VRE* function given in Equation 8.5. Therefore, the objective function to minimize can be written as follows:

$$\mathcal{F}(k) = \min[VRE(k)]. \tag{8.8}$$

8.5.4 Genetic Operators and Termination Criterion

Genetic operators use the objective and selection functions to apply some modifications on selected individuals to produce offspring of the next generation. This process provides new possibilities in the solution space by combining the fittest chromosomes and passing superior genes to the next generation. There are numerous implementations of genetic operators. For instance, there are dozens of possible crossover and mutation operators that have been developed in recent years.

A heuristic crossover generating a random number v from a uniform distribution and doing an exchange of the parents' genes (X and Y) on the offspring genes (X and Y) is used in this application. The main characteristic of this type of crossover is that it utilizes the fitness information. Offspring are created using the following equation:

$$\begin{aligned} X' &= X + v(X - Y) \\ Y' &= X, \end{aligned} \tag{8.9}$$

where X is assumed to perform better than Y, in terms of objective function. Heuristic crossover introduces a *feasibility* function Fs, defined by:

$$Fs(X') = \begin{cases} 1 \text{ if } a_i \leq x'_i \leq b_i \quad \forall i \\ 0 \text{ otherwise,} \end{cases}$$

where $x_i^{'}$ are the components in N-dimensional space, of X' with *i=1,...,N*. The Fs function controls the generation of a new solution using Equation 8.9. In fact, when $Fs(X')$ equals 0, a new random number v is generated in order to create the offspring.

Mutation operators lead to small random changes of the individual components in an attempt to explore more regions of the solution space [29]. The principle of a non-uniform mutation used here consists of randomly selecting one component x_k of an individual and setting it equal to a non-uniform random number, otherwise, the original values of components are maintained. The new component, x_k', is given by:

$$x_k' = \begin{cases} x_k + (b_k - x_k) f(Gen) \text{ if } u_1 < 0.5 \\ x_k - (a_k + x_k) f(Gen) \text{ if } u_1 \geq 0.5 \end{cases} \tag{8.10}$$

where the function *f(Gen)* is given by:

$$f(Gen) = \left(u_2 \left(1 - \frac{Gen}{Gen_{max}} \right) \right)^t, \tag{8.11}$$

u_1, u_2 are uniform random numbers selected within (0,1), t is a shape parameter, *Gen* the current generation and Gen_{max} the maximum number of generations. We have shown in [123] that the use of the heuristic crossover and the non-uniform mutation [65] are suited for the reduction of additive noise.

When a certain number of predetermined generations is reached, the evolution process is terminated. The fittest individual, which corresponds to the best set of weights or optimal axes, is then used to project the noisy data. Then, the "genetically modified" MFCCs and CDCs are used as enhanced features for the testing phase of the recognition process.

8.6 Experiments and Results

8.6.1 Speech Material

The TIMIT corpus [133] is used to evaluate the KLT-VRE-GA approach. Timit contains broadband recordings of 6300 sentences: 10 sentences spoken by each of 630 speakers from 8 major dialect regions of the United States, each reading 10 phonetically rich sentences. All train subsets of the TIMIT database are used to train the models and the test sub-directories are used to evaluate the recognition systems.

Table 8.1 Genetic parameters used in the application.

Parameter	Parameter Value
Number of generations	500
Population size	200
Probability of selecting the best g_i	0.10
Heuristic crossover rate	0.35
Multi-Non-Uniform Mutation rate	0.05
Number of runs	70

8.6.2 Recognition Platform

The HTK HMM-based speech recognition system described in [66] has been used throughout all experiments. The HTK toolkit was designed to implement HMMs with any numbers of state and mixture components. It also allows the creation of complex model topologies to suit a variety of speech recognition applications. All the tests are performed using N-mixture ($N = 1, 2, 4, 8$) Gaussian HMMs with tri-phone models.

8.6.3 Tests & Results

To simulate a noisy environment, various noises are added artificially to the clean speech at different SNR levels varying from 16 dB to -4 dB. The reference models are created using clean speech. Four different sets of experiments are designed. The first set concerns the baseline system in which 12 MFCCs are calculated over a 30-msec Hamming window. The normalized log energy is replaced by the mid-external energy of the ear extracted by means of the Caelen's model. The dynamic features that are the first and second derivatives of MFCCs are also included. Furthermore, in order to compare the KLT-VRE-GA system with a recognizer using a well-established noise-reduction technique, the mean normalization of MFCCs (CMN) is applied to the 12-dimensional MFCC vector. The second set involves the well-known state-of-the-art eigen-decomposition method, the one based on the KLT applied in the Mel-frequency space (c.f. Chapter 3). The third set of tests carries out speech recognition using the evolutionary-based eigen-decomposition (KLT-VRE-GA) method. In the last set of tests, the static vector composed of the 36 MFCCs and their derivatives is expanded by adding the 7 CDCs and the mid external energy of the ear, to form a 44-dimensional vector upon which the baseline multi-stream HMMs were trained.

The values of the GA parameters used in our experiments are given in Table 8.1. To realize the compromise between speed and accuracy, a population of 200 individuals is generated for each axis. The objective function stabilizes (no improvement is noticed) after approximately 300 generations. In order to insure the convergence in all situations, the maximum number of generations was finally fixed at 500.

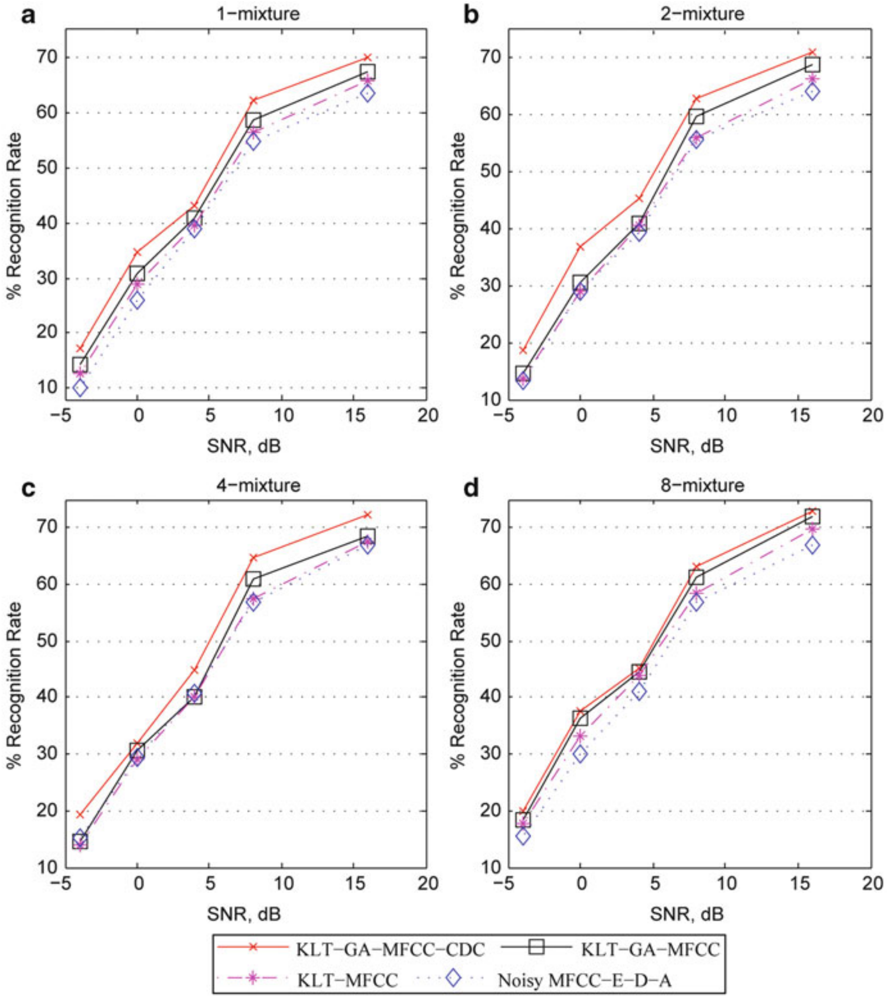

Fig. 8.2 Comparisons of the percentage of word recognition ($\%C_{wrd}$) of HTK ASR systems using N-mixture ($N = 1, 2, 4, 8$) triphones and TIMIT database corrupted by additive car noise: the first ASR system uses MFCCs and their first and second derivatives with mean normalization. The second system uses the KLT normalized MFCCs using the VRE objective function, the third includes the KLT-GA-based front-end applied to MFCCs, and finally the fourth ASR system applies the KLT-GA to the CDCs and MN-MFCCs.

As shown in Figure 8.2, the system including the KLT-GA based front-end achieves higher accuracies compared to the baseline system dealing with noisy normalized MFCCs. This improvement is observed for all SNR values and varies within a range of 3% and 8%. The KLT-GA-CDC system which combines the MFCCs, their first and second derivatives and CDCs outperform the other systems for all values of SNR.

8.7 Summary

This chapter has presented a promising approach advocating the usefulness of evolutionary-based subspace filtering to complement conventional ASR systems meant to tackle the challenge of noise robustness. In fact, the combined effects of subspace filtering optimized by GAs and knowledge gained from measuring the auditory physiological responses to speech stimuli may provide more robustness to speech recognition. It should be noted that such a soft computing technique is less complex than many other robust techniques that need to either model or compensate for noise. Many other directions remain open. The research on acoustic-phonetic features in ASR should benefit more from the knowledge related to the auditory system. A promising way is to modify the basic preprocessing technique to integrate phonetic knowledge directly. Distinctive cues can be learned by an arbitrary function approximator such as an artificial neural network (ANNs), yet another soft-computing technique. The training of ANNs on acoustic distinctive feature labels will permit us to gain a more effective representation of the acoustic-phonetic mapping function. Using this approach avoids the noise estimation process that requires a speech/non-speech pre-classification, which could not be accurate for low SNRs.