Robust Text-Independent Speaker Verification Using Genetic Programming

Peter Day and Asoke K. Nandi, *Senior Member, IEEE*

***Abstract*—Robust automatic speaker verification has become increasingly desirable in recent years with the growing trend toward remote security verification procedures for telephone banking, biometric security measures and similar applications. While many approaches have been applied to this problem, genetic programming offers inherent feature selection and solutions that can be meaningfully analyzed, making it well suited to this task. This paper introduces a genetic programming system to evolve programs capable of speaker verification and evaluates its performance with the publicly available TIMIT corpora. We also show the effect of a simulated telephone network on classification results which highlights the principal advantage, namely robustness to both additive and convolutive noise.**

***Index Terms*—Genetic programming (GP), speaker verification.**

I. Introduction

CURRENT automatic speaker recognition (ASR) systems generally fall into one of two categories: automatic speaker identification (ASI) systems aim to answer the question "who is the speaker?", while the aim of an automatic speaker verification (ASV) system is to answer the question "is the speaker who they claim to be?". These problems have enjoyed research interest both independently and jointly over recent years, and while they share similar strategies, their applications are quite different. While most ASI systems deal with a known (closed) set of speakers, the challenge of an ASV system is to establish the presence of a speaker within an unknown (open) set of speakers. In this paper we primarily deal with the latter (ASV) problem, although, as with most proposed systems, the techniques demonstrated could be easily adapted to the ASI problem.

While many techniques have been applied to the problem of ASR, genetic programming (GP), to the authors knowledge, has not been applied previously. Promising results have been shown using other Artificial Intelligence (AI) methods, such as support vector machines (SVM) and artificial neural networks (ANN); this will be discussed further in the following paragraphs. This paper intends to demonstrate the advantages of GP in generating robust ASV solutions that are tailored to individual speakers.

Manuscript received February 1, 2005; revised November 27, 2005. This work is supported by the EPSRC, U.K. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Bayya Yegnanarayana.

The authors are with the Department of Electrical Engineering and Electronics, The University of Liverpool, Liverpool L69 3GJ, U.K (e-mail: peterday@liv.ac.uk; aknandi@liv.ac.uk).

Digital Object Identifier 10.1109/TASL.2006.876765

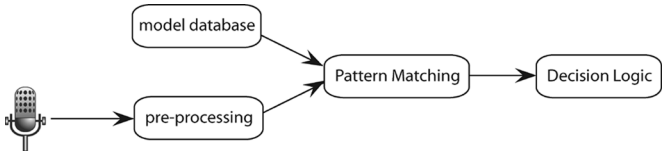

Fig. 1. Typical ASR system.

A. Overview of Existing and Proposed Systems

Most proposed ASR systems, whether their primary goal is ASV or ASI, share similar strategies. The typical process involves: some form of preprocessing of the data (silence removal) and feature extraction, followed by some form of speaker modelling to estimate class dependent feature distributions (see Fig. 1). A comprehensive overview can be found in [1]. Adopting this strategy the ASR problem can be further divided into the two problem domains of:

1) preprocessing, feature generation and selection;
2) speaker modelling and matching.

Features and Feature Selection: The choice of features in any proposed ASR system is of primary concern [2], [3], because if the feature set does not yield sufficient information then trying to estimate class dependent feature distributions is futile. As with all feature dependant classification problems, the ideal feature set should give features that have high inter-class variance and low intra-class variability. Ideally the selected features should also be as independent of each other as possible in order to minimise redundancy.

There is a huge amount of information present in a speech signal and speech can be described as having a number of different *levels* of information. At the top level, we have *lexical and syntactic features*, such as language use and sentence construction. These require a lot of intelligence to understand and interpret, and automating this process is the aim of the speech recognition research. Recently, lexical and syntactic features have been used in ASR systems (see [4] and [5]), these systems tend to be more resilient to noisy transmission paths, however there is a high computational cost as the proposed systems are required to interpret the text at some level, so the systems cannot be considered truly independent of text.

Below these are *prosodic features*, which represent information such as: intonation, stress, and rhythm of speech. Further below these are *Phonetic features* which represent the "sound" of individual syllables, and at the most basic level we have *low-level acoustic features*, which generally give information on the system that creates the sound, such as the speakers' vocal tract.

It is likely that the human brain uses a combination of these levels of information when identifying a speaker. However this does not necessarily imply that a machine needs to understand the syntactic meaning of a sentence in order to correctly identify an individual. If we consider these levels of information as indicators solely of speaker identification, then the lexical and syntactic features are largely redundant (it is not usually possible to determine who is speaking from a transcript of something they have said). Similarly, while *Prosodic* and *Phonetic* features may give speaker dependent information, it is also likely to be text or time dependent and thus not obviously suited to this task. Information solely about *how* the sound is produced (from low level acoustic features) should give enough information to identify accurately a speaker as this is naturally speaker dependent and independent of text.

Most previous works relied on the use of *low-level acoustic features*. Mel-frequency cepstral coefficients (MFCCs) have been particularly popular in recent years as they give a highly compact representation of the spectral envelope of a sound. Many proposed systems have relied solely on this data and good results have been reported. Linear prediction coefficients and the closely related Line Spectrum Pairs have also been popular, as have the related perceptual linear prediction values which have been shown to be more robust in noisy environments. Other work also combines multiple feature sets, such as pitch and MFCCs in the case of [6], and this approach has also been shown to give a more robust solution.

It is both conceivable and probable that different features have a different level of *importance* in characterizing different voices. For instance, if we meet someone with an unusually high-pitched voice, this is likely to become the primary information that we use when recognizing or describing that individuals voice—it is more obvious than more subtle characteristics such as intonation or accent (and these characteristics are redundant). While in this example the pitch may be enough to characterise the voice, in other cases, where the pitch of an individual voice is, by definition, more "*normal*", this information is likely to be meaningless and other features (those that do distinguish the individual voice from the "*normal*") are required. It is easier to remember or describe a voice by deducing how it differs from our experience of what is normal. While successful classification or verification for the majority of voices has been demonstrated using a shared feature set, some voices are harder to classify robustly than others [7]. We propose that these more difficult to classify voices are a result of a limited shared feature set. If a system finds it hard to classify a subset of the voices it analyses it follows that they are similar, at least as far as the limited feature set is concerned. Following the assumption that the ideal set of features to distinguish an individual voice from all others (i.e., a feature set that *characterises* the voice) will vary between different voices, it can be seen that an ideal strategy would allow some level of feature selection on a *per voice* basis.

Speaker Modelling: The speaker modelling stage of the process varies more in the literature. Speaker modelling takes place when an individual is enrolled into an ASR system with the aim of defining a model (usually feature distribution values) that *characterises* the individual. The two most popular methods in previous work are vector quantization (VQ) and Gaussian mixture models (GMM). Other techniques such as mono-Gaussian models [8], decision trees [9], SVM [10], [11] and ANN [12] have also been applied, and a good comparison and overview can be found in [13] and [14].

In the VQ method [15], speaker models are formed by clustering the speaker's feature vectors in K non overlapping clusters. Each cluster is represented by its centroid (average vector) and the resulting collection of K centroids is referred to as its code book and serves as a model for the speaker. The two considerations when using this method are: what method to use to perform the clustering and what size codebook to use. There is much literature which discusses these issues and a comprehensive overview can be found in [2].

The GMM method differs from the VQ method in that it is a parametric method: A GMM consists of K Gaussian distributions parameterised by their *a priori* probabilities, mean vectors and covariance matrices. The parameters are typically estimated by maximum likelihood estimation. A typical implementation of this system for ASR can be found in [16].

Previous work also includes classifier ensembles [17], where multiple modelling techniques are used, chosen according to their suitability to a given feature set. Decisions are made dependent on the values given by several independent classifiers.

While good results have been reported using a variety of different systems, most systems suffer heavily if the bandwidth is limited or the signal is transmitted over a noisy transmission path (such as a telephone network). This is largely blamed on the features used being impaired by noise.

II. Genetic Programming

GP is an *automatic programming* technique based on the Darwinian concepts of evolution. While techniques such as ANN have demonstrated good results in the wider field of machine learning (and with speaker verification), GP reflects nature on a larger scale and has been shown to be highly adaptable and offers innovative solutions. GP has been applied to other classification tasks with great success (see [18], for example) and feature selection is inherent in the process of "evolving" a classifier, making it well suited to classification tasks as feature selection is often done independently of classification—possibly by a precursor to GP, a genetic algorithm (GA).

The premise behind GP and all other evolutionary techniques is survival of the fittest. Initially randomly created individuals (candidate solutions) are rated according to their ability to do a given task (in this case verify a voice), with the fittest being selected for further processing (such as breeding to create new individuals). Each candidate solution in GP is a computer program, and so the GP process can be considered a search trough the *program space* for a program capable of performing the desired task. In order for the search process to be effective, a fitness function must be used in order to guide the direction of evolution, and typically the fitness function measures how accurately each candidate solution performs a given task.

When GP is applied to classification problems, the process generally generates a program tree composed of mathematical functions (both linear and nonlinear) and input features, so it could be considered a nonlinear combination of features. The

aim of the evolutionary process is that output of the program should give an indication of the class of the input data.

The use of GP for ASV allows the evolution of programs *tailored* to individuals. This implies that the features used will also be specific to the individuals, which we propose will provide a more robust solution. This also allows solutions to be tailored to different environments as the process does not rely on a single feature set, combinations of features can be inherently selected that perform well for an individual under given circumstances (such as over a telephone line). This system could also be readily extended to include new features if they are required with minimal impact to the evolution and classification process. While the computational cost for evolving a good solution is relatively high, lexical, and syntactic features have been excluded so that the evaluation cost is low.

It is beyond the scope of this paper to accurately describe the GP process; readers unfamiliar with this process should refer to [19] or [20].

A. A Fitness Function

The fitness function is the principal concern when establishing a GP process, as this states the problem and thus determines the direction of evolution. In the case of ASV we can simply specify that the ideal program should output a *one* if the speaker is detected and a *zero* otherwise. The simplest way of doing this is to base a fitness measure on the difference between the desired output of the GP program and the measured output for a given set of inputs with an equation as follows:

$$\text{fitness} = \sum_{n=1}^{N} (\text{desired}_n - \text{given}_n)^2 \tag{1}$$

where N is the number of training cases, desired_n and given_n are the desired result and measured result for training case n respectively. The ideal fitness will be zero (i.e., there is no difference between the given values and the desired values—the program is *ideal*).

While this fitness function may suffice, it has not proved robust for complex classification problems such as ASV in the authors' experience. With this strategy a good fitness value can be achieved by individuals that perform very well with a subset of training cases, but poorly with other. This is clearly not acceptable for classification problems, so measures must be taken to promote convergence at solutions where *all* training cases are correctly classified. i.e., it is better to classify more cases correctly than make fewer cases closer to the desired value.

In order to drive the poorly performing fitness cases (outliers) closer to the desired value we introduce a more complex fitness function. First, we simplify the problem by scaling outputs given by each GP program to the range 0–1. Next, each individual is assigned a binary value (p_n) for each of the training cases (n), this value indicates whether the output of the program for the training case is within a given range (k) of the desired value, thus

$$\begin{aligned} \text{if } |\text{desired}_n - \text{given}_n| < k, \quad & p_n = 1 \\ \text{otherwise} \quad & p_n = 0 \end{aligned} \tag{2}$$

where k is a dynamic constant that changes during the evolution process. Initially (generation 0) the value is set to 0.5, so that it determines whether the output of the individual is within the limit of data correctly classified. If an individual succeeds in having all its training data within this range of the expected value (i.e., all p_n values are equal to one) the value of k is reduced to the mean error of that individual for the next generation. This evaluation is carried out in every generation of the evolution process. and has the effect of first ensuring that all training cases are classified correctly, then reducing the intra-class variance.

Considering the p_n values, the resultant raw fitness (f_r) function is then given by

$$f_r = N + \sum_{n=1}^{N} \left((\text{desired}_n - \text{given}_n)^2 - p_n \right). \tag{3}$$

The inclusion of the number of training cases (N) is to ensure that the optimal fitness is *zero*. While this is not essential, it is used to simplify the stopping criteria used.

Finally it is also important to add some parsimony pressure [21] to prevent code bloat. This process is well documented and established, and the simplest implementation is to penalise individuals according to their size, thus

$$f_p = f_r + cS \tag{4}$$

where f_p gives the fitness of the individual including parsimony pressure, c is a constant that dictates the penalty associated with size (typically 0.001), and S is the size of the individual (i.e., the total number of nodes).

In our experiments, we use this strategy in association with a dynamic tree depth limit [22] that sets an initially short maximum depth for individuals when the run is started. Trees are only allowed to exceed this limit if their overall fitness is better than the previous best fitness. If this is the case, then the maximum size for allowed trees is increased to the depth of the new best of run individuals depth.

While this fitness function does steer the direction of evolution towards solutions that classify more solutions correctly, the process still suffers from suboptimal convergence (i.e., solutions which struggle to classify a subset of training examples). In the following text, we introduce some new concepts that further promote convergence at solutions which perform well with *all* training cases.

B. Comparative Partner Selection (CPS)

In GP, the reason often cited for suboptimal convergence is the tendency toward diminishing genetic diversity as the evolutionary process (selection of the fittest) transpires. While arriving at the correct or optimal solution is clearly dependent on individuals that resemble this solution being selected for further processing, this also increases the likelihood of convergence at a suboptimal solution. The aim of CPS is to promote populations that are capable of performing equally well with all training cases and hence reduce the probability of arriving at

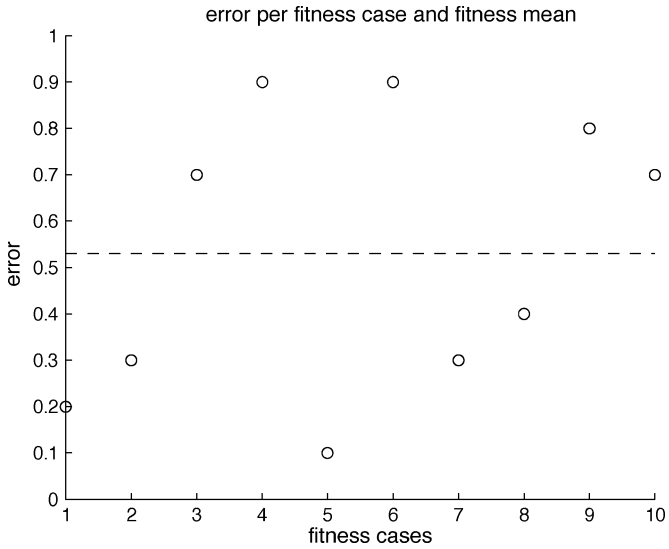

Fig. 2. Generation of a *binary string fitness measure* string for ten fitness cases. The mean is represented by the dashed line, hence all cases above the line will be represented by a *zero* (representing a weakness), and all others represented by a *one* (representing a strength). The resulting fitness string in this case is 1100101100.

a suboptimal solution (which also leads to smaller population sizes being required) and an increase in the speed of evolution.

CPS is reliant on the method of assigning each individual in a generation a binary string fitness measure (BSFM) in addition to the overall fitness value that is required in all evolutionary methods. The BSFM indicates the strengths and weaknesses of the individual and while this appears at first to be a similar concept to the p_n values discussed previously (2), their applications are quite different. The BSFM is used to give an indication of the behavior (or *phenotype*) of an individual.

The process for assigning an individual a BSFM is trivial: a *one* represents a strength and a *zero* represents a weakness, i.e., any training case where the error is less than or equal to the mean error for that individual results in a *one*, any training case that performs worse than the mean is indicated by a *zero*. Consequently an individual that gives the "correct" answer will have a fitness string that comprises solely of ones. An example of this process is shown in Fig. 2. The length of the fitness string will be dependent on the number of training cases used. In the case of evolving an ASV program the BSFM will represent how well classified each of the training voices are, i.e., which examples the individual can classify most accurately, and which voices it is not as good at classifying.

It is our proposal that it is beneficial to the population as a whole to maintain the *behavioral* diversity of the population, such that the population performs equally well with all training cases. Furthermore, we suggest that strengths and weaknesses (and hence the binary string described above) can be used as a simple measure of an individuals' behavior. Following these lines it can be seen that implementing an "opposites attract" strategy when selecting parents for crossover operations will maintain the ability of the population to solve all fitness cases equally, so that a population does not reach a state where it contains no genetic material capable of solving a subset of training cases.

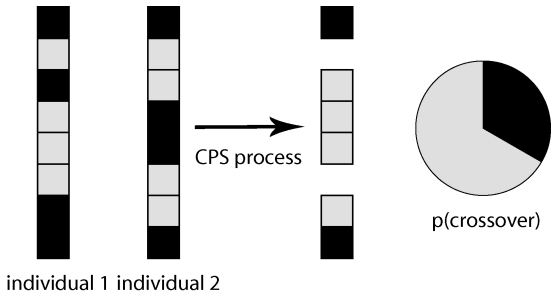

Fig. 3. Example of the CPS process for two individuals with 8-bit BSFMs. The lighter squares represent *ones*, while the darker squares represent *zeros*. In this case, the probability of crossover is 2/3.

A probability of crossover that increases the chance of crossover if one individual is strong in an area that the other is not, and decreases the chance if they are both weak in the same area can be calculated using simple (and computationally inexpensive) logical operations

$$p(\text{crossover}) = \frac{\sum \text{XOR}(f_1, f_2)}{\sum \text{NAND}(f_1, f_2)} \tag{5}$$

where f_n is the binary fitness string of individual n and $\sum$ represents the summation of each bit in the binary string. This process can be seen clearly in Fig. 3.

Implementation of this strategy is relatively straightforward.

- An individual is selected according to a standard GP method (such as the roulette system), based solely on the fitness *value*.
- A second individual is selected in the same manner.
- The probability of the two individuals producing offspring is calculated.
- It is decided (by generating a random number) whether the two individuals will reproduce or not.
- If the individuals do reproduce then a new starting individual is selected, otherwise another secondary individual is selected and the process is repeated.

Clearly, it is possible that one individual will not find a suitable individual to reproduce with, if this is the case, after a predetermined search time (such as meeting 50% of the population) a partner is selected for crossover without considering the fitness strings.

In terms of nature, the fitness values are used to determine the probability of two individuals meeting and the fitness strings are used to determine the probability of two individuals choosing to mate.

Work has shown that without CPS, evolving programs capable of performing robust ASV has a poor success rate as the evolutionary process *usually* converges at solutions which perform poorly with a subset of the training cases. Using CPS in conjunction with the fitness function introduced the system is intolerant of convergence to suboptimal solutions and hence provides useful classifiers.

C. Multiple Populations—The Island Model

It has been shown in previous literature [23] that the introduction of the island model to the GP process offers signifi-

cant advantages in the speed of evolution and avoiding convergence at suboptimal solutions, and makes the parallelisation of the process simple. In this paper we use an asymmetric island model, similar to the one discussed in [24]. However, using dynamically chosen migration routes we additionally promote migration between genetically different *islands*. This is accomplished by evaluating each *island* in terms of strengths and weaknesses of its current generation. Each population is assigned a fitness string representing how successfully the individuals making up the population solve each of the test cases. The individuals error values are weighted according to their overall fitness to stop the worst individuals in the population dominating the overall impression of strengths and weaknesses

$$\text{popFitnessString}_n = \sum_{i=1}^{I} \frac{\text{error}_{n,i}}{\text{fitness}_{n,i}} \tag{6}$$

where I is the total number of individuals in the island population, and n represents the fitness case.

A BSFM is then assigned to the island in the same manner as previously described for individuals (see Fig. 2). The CPS method (5) is then used to establish "migration" routes, although it is referred to as *comparative island selection* (CIS) in order to avoid confusion. In our experiments each island has *three* migration routes at the end of every generation, so the process is as follows.

- Three pools of individuals are selected using a roulette system weighted according to fitness (these are left "ready to be exported" to other islands).
- An island is selected at random; if it has any "ready to be exported" pools, the CPS evaluation is performed (to determine probability of migration).
- It is decided according to probability of migration whether the migration will take place.
- If the migration takes place, individuals from one of the pools of the selected island are imported into the next generation, otherwise another island is selected. Migration is not symmetrical (each island only selects islands from which to import individuals).
- Island selection and evaluation process is repeated until three pools have been imported into the next generation.

If a suitable migration route cannot be found (after performing island selection and CIS evaluation with all potential islands) one of pools of individuals from the "ready to be exported" set is incorporated back into the next generation. It is possible for one population to set up multiple migration routes with a single island (assuming there are individuals left to import), meaning that more individuals will migrate from that island. The migration rate for all experiments in this paper is set at 1% per migration route (3% of individuals in total will have migrated). This proportion of migration is consistent with previous works in GP and has proved sufficient to promote diversity.

The effect of multiple populations in solving the ASV problem is not nearly as dramatic as the effect of CPS, in fact the same performance can be achieved by using a single population with around 20% more *total* individuals. However, this strategy does have a dramatic effect on the ability to distribute the process (i.e., run the evolution in parallel over a cluster of computers), making it almost trivial.

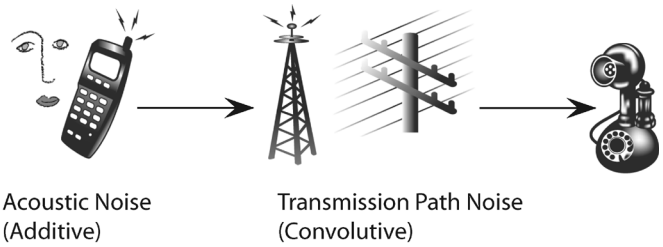

Fig. 4. Model of typical voice transmission over a telephone network.

TABLE I
DESCRIPTION OF DATASETS USED

Name	
TIMIT	Standard corpora
F1-TIMIT	Filtered through *filter 1* (see text)
F2-TIMIT	Filtered through *filter 2* (see text)
F3-TIMIT	Filtered through *filter 3* (see text)
TIMIT-8	corpora down-sampled to 8kHz
F1-TIMIT-8	down-sampled to 8kHz and filtered through *filter 1*
F2-TIMIT-8	down-sampled to 8kHz and filtered through *filter 2*
F3-TIMIT-8	down-sampled to 8kHz and filtered through *filter 3*

III. TEST DATA

A. TIMIT Corpora

The TIMIT corpora [25], [26] is used in this paper due to its high number of speakers and its use in other literature making it useful in comparison with other proposed methods. This corpora contains 630 speakers (438 male and 192 female) each speaking ten sentences, there are two sentences that are spoken by all speakers and the remaining eight are selected randomly from a large database. The speech signal is recorded through a high quality microphone with a sampling frequency of 16 kHz in a quiet environment.

B. Noisy TIMIT

One of the principal applications of ASV systems is remotely confirming the identity of a person for reasons of security such as telephone banking, so we must consider the transmission of the speech over telephone network when finding a solution; an example of such a system is shown in Fig. 4.

In order to create a more realistic training and testing environment, several datasets were derived from the original corpora (these are outlined on Table I). First, a dataset was derived by down-sampling the original corpora to 8 kHz (*TIMIT-8*). Three different simulations of typical phone transmissions were created through the use of three different filters, all three filters include both additive and convolutive noise. *Filter 1* simulates a plain old telephone system (POTS), *Filter 2* simulates a GSM mobile network and *Filter 3* simulates a more complex mobile and POTS systems (similar to the model shown in Fig. 4). *Filter 3* is reserved purely for Testing purposes and is not included in any of the training sets.

C. Training and Test Sets

Twenty-five speakers were chosen at random (19 male, six female) to represent the to-be-verified (TBV) individuals. A further 45 speakers (32 male, 13 female) were chosen at random to represent the "impostors" in the training process. The remaining speakers were used in the testing procedure.

Each speaker in the TIMIT corpora has around 30 s of speech. We divide this into two equal halves for each of the TBV individuals for training and testing respectively. This gives 15 s of TBV speech for the training process, this is coupled with 1 s of randomly selected speech from each of the 45 "impostor" individuals (a total of one minute training speech).

In order to increase the initial rate of evolution, only a small subset (5 s TBV and 10 s of "impostor" speech) is introduced at the beginning of the run. This allows individuals to be evaluated more quickly yet there is enough data to give an indication of individuals that are likely to be successful. More training data is added every time the k value from (2) is reduced (i.e., when the best individual is correctly classifying all current training examples). Additional training data is not added to all islands simultaneously. If one island has additional training cases when calculating the probability of migration tunnels being formed, then only the first N bits of the populations BSFM are considered (where N is the least number of training cases being used by either island). i.e., if Island *A* has 15 training cases and Island *B* has 25, the CIS strategy is performed between the population fitness string of Island *A* and the first 15 bits of the population fitness string for Island *B*.

We also experiment using multiple training sets (e.g. TIMIT and F1-TIMIT), with the intention of evolving a classifier that is robust to multiple environments. This strategy also has the benefit of more training data being available.

The test set comprises of the remaining 15 s of the TBV individuals data and 100 single second samples randomly chosen from the corpora (not including the training set of "impostor" speakers). Each of the best-of run individuals for each of the TBV individuals is evaluated with all datasets.

D. Feature Generation

Audio Descriptors are important in many fields (such as speech recognition, voice synthesis and recognition, sound classification, etc.) and many descriptors have been suggested (see [27]–[29] for examples). By using GP, one has the advantage of allowing the process to select the most important features and discard any unnecessary ones, hence many descriptors are generated, but the initial stages are the same for all.

First, the speech signals are sliced into roughly 1-s "portions," each of these portions is treated as a separate input signal. These portions are further divided into windows of 1024 samples (with a 10% overlap) for data sampled at 16 kHz and 512 samples for data sampled at 8-kHz data (approximately 0.06 s), and for each of these windows the following descriptors are generated, and their mean and variance are recorded as the input descriptors.

The range of features included is purposely large, as while some of the features do not relate directly to speaker individuality (such as active sound level), they may be of some use in evolving more complex programs.

TABLE II
THE FUNCTION POOL

name	arity	function
sin	1	$sin(x)$
log	1	$log(x)$
ln	1	$ln(x)$
logsig	1	$1/(1+exp(-x))$
tansig	1	$2/(1+exp(-2x))-1$
round	1	round x to nearest int
ceil	1	round up x to nearest int
floor	1	round down x to nearest int
average	2	$mean(x_1, x_2)$
exp	1	$exp(x)$
log10	1	$log_{10}(x)$
lt	2	is x_1 less than x_2
gt	2	is x_1 more than x_2
eq	2	does x_1 equal x_2
neq	2	does x_1 differ from x_2
pow	2	$x_1^{x_2}$
sqrt	1	$\sqrt{x}$
nroot	2	$\sqrt[x_2]{x_1}$
plus	2	$x_1 + x_2$
minus	2	$x_1 - x_2$
multiply	2	$x_1 x_2$
divide	2	$x_1$if $x_2 = 0$, else x_1/x_2
and	2	are x_1 AND $x_2 \geq 1$
not	1	if $x \geq 1$, 1, else 0
or	2	are x_1 OR $x_2 \geq 1$
if	3	$x_2$if $x_1 \geq 1$else x_3

Mel-Frequency Cepstral Coefficients: Mel-frequency cepstral coefficients have usually been associated with speech analysis and processing, and can be used as a compact representation of the spectral envelope. These coefficients are described as perceptual features as they are chosen to mirror how our brain is understood to comprehend sound.

Delta MFCCs (which are believed to indicate the speech rate) are also included as these have shown good results in previous ASR literature. We consider both sets of coefficients up to the tenth order.

Linear Prediction Coefficients (LPC): Linear prediction, the process of predicting future sample values of a digital signal from a linear system, has been used with success in speech processing and coding, and LPC are believed to give very accurate formant information of acoustic signals. We consider the LPC up to the 14th order (excluding the zeroth-order coefficient).

Perceptual Linear Prediction Coefficients (PLP): PLP [30] are highly related to LPC, but take advantage of psychoacoustic principles and consequently are generally more robust in noisy environments. We consider the first nine PLP.

Spectral Descriptors: Further descriptors can be derived that give other information on the spectral envelope.

- Spectral Flatness: This is a ratio between the geometric mean and the arithmetic mean of the spectrum. This descriptor gives an idea of the flatness; the higher it is, the closer to white noise the signal is.
- Spectral Centroid: the centre of gravity of the spectrum
- Spectral skewness: the third-order central moment gives an indication about the symmetry of the spectra.
- Spectral kurtosis: the fourth-order moment gives further indication of the spectra.

TABLE III
RUN PARAMETERS

Parameter	Value(s)	
Genetic Operators	Type	Percentage
	one-offspring Crossover (internal points)	50%
	Terminal Crossover (non PNV)	9%
	PNV Crossover	9%
	Gaussian Mutation of PNV	19%
	Standard Mutation	3%
	Reproduction	10%
Population Size	12 islands of 5000 individuals (60,000 total)	
Population Migration	3% asynchronous Migration (see text)	
Stop Conditions	350 generations or when all $P_n = 1$ and $k < 0.01$ (see eq 2)	
Elitism	Half-Elitist (see text)	
Parent Selection	Roulette Method with CPS	

- Spectral Crossings: Spectral crossings (at a set threshold) can be used to determine how many strong peaks there are in a spectra.
- Spectral Peaks (index and magnitude), the largest five are considered (giving ten descriptors in total).

Temporal Descriptors:

- Zero Crossing Rate: the number of times the signal changes from negative to positive values.
- Active sound level: The amount of acoustic energy present.
- Variances in all the descriptors mentioned between frames (i.e., their variance) can also be considered temporal descriptors.

In total there are 60 descriptors, each with a mean and variance over the 1-s frame, leading to 120 possible input descriptors to the GP individual. In addition to the descriptors, the possible terminal pool also includes integers, fractions, a random number generator, and other constants such as π.

The final inclusion is the perturbable numerical value (PNV) [31]: A value that is initially created at random (within the range -1 to 1), but can be changed, or perturbed, during the evolutionary process. Values are perturbed through a discrete operator, and the *to be perturbed* value is considered the mean of a Gaussian distribution (with standard deviation of 0.25), the amount the value is perturbed is determined by this distribution such that most perturbations will be small in magnitude.

E. Function Pool

The functions available to GP individuals are shown in Table II. Both linear and nonlinear functions are included in addition to logical operators. The inclusion of logical operators allows the programs to use complex decision logic. Each function also has an *arity*, this is simply the number of input values the function requires.

Using the above descriptors and functions, a candidate solution might look like the tree shown in Fig. 5. While successful solutions are generally a lot more complex and consequently hard to meaningfully analyze, this tree gives an impression of the nature of a solution.

F. Additional Run Parameters

An overview of the run parameters can be seen in Table III. Due to the nature of GP, most of the parameters are decided by trial and error, or "rule of thumb" approximations. The values

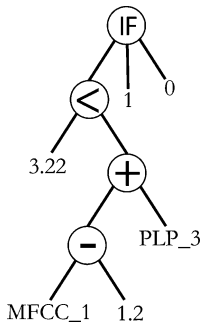

Fig. 5. Potential candidate solution given the descriptors and functions used.

used in this paper are largely based on previous experience and performing a large number of trial and error studies with the ASV problem in particular and have been chosen to give a good chance of convergence at robust solutions. While smaller population sizes may result in equally good solutions (and require considerably less computational cost), the chance of arriving at *good* solutions is low, the population sizes used in this paper are chosen to provide a robust solution, at possibly higher computational costs.

In addition to parameters previously discussed, a half-elitist strategy is used. This strategy means that a new generation is made up of half completely new individuals (selected proportionate to fitness) and the remaining half is comprised of the best individuals from the previous generation and the new generation. The majority of these parameters are *typical* values for GP, and have been arrived at empirically.

IV. RESULTS

A summary of results can be seen in Table IV, the results are given as percentage of test cases correctly classified (for all 25 TBV individuals). The classification process simply takes the output of the GP-generated program and the range of outputs given by program for the training data; if the output of the program is closer to the range of the TBV individuals training data it is classified thus, otherwise it is classified as "impostor". A more decision making process could be adopted, but this simple routine proves sufficient.

When the training and test sets used are taken from the standard TIMIT corpora, the correct classification rate is similar to previous published results. However, the advantages of the

TABLE IV
SUMMARY OF CLASSIFICATION RESULTS FOR THE BEST OF RUN INDIVIDUALS

	Test Set								
Training Set	TIMIT	F1-TIMIT	F2-TIMIT	F3-TIMIT	TIMIT-8	F1-TIMIT-8	F2-TIMIT-8	F3-TIMIT-8	Overall
TIMIT	99.5%	81.2%	77.8%	74.5%	79.0%	78.4%	74.1%	72.0%	79.6%
F1-TIMIT	92.1%	93.5%	89.3%	91.8%	89.1%	83.7%	81.4%	79.8%	87.6%
F2-TIMIT	98.8%	92.4%	97.9%	93.3%	90.4%	89.9%	90.8%	88.5%	92.8%
TIMIT-8	98.6%	88.7%	92.7%	91.1%	97.9%	82.0%	87.8%	90.2%	91.1%
F1-TIMIT-8	98.1%	93.1%	88.9%	90.0%	95.5%	93.7%	90.7%	89.8%	92.5%
F2-TIMIT-8	98.0%	92.8%	96.7%	93.8%	96.2%	91.8%	92.6%	90.3%	94.0%
TIMIT & F1-TIMIT	99.2%	97.3%	93.8%	94.0%	91.6%	91.2%	88.8%	91.6%	93.4%
TIMIT & F2-TIMIT-8	99.8%	96.1%	97.8%	95.7%	98.1%	89.6%	94.6%	93.8%	95.7%
TIMIT & F1-TIMIT & F2-TIMIT-8	99.7%	97.1%	95.8%	96.2%	98.9%	94.7%	90.9%	94.0%	97.0%

TABLE V
RESULTS USING GMM GENERATED SPEAKER MODEL

Feature Set	clean environment (TIMIT)	same noisy environment (F1-TIMIT)	different noisy environment (F1-TIMIT and F3-TIMIT)
PLP	98.2%	90.2%	78.1%
MFCC	99.6%	86.3%	74.0%
PLP + MFCC	99.1%	93%	83.9%

process introduced in this paper are more clear when handling *noisy* data. Indeed, when the training set is comprised of both clean and noisy data, the resulting program achieves up to 97% correct verification across all testing environments. This is considerably better than other reported results with similar experiments in noisy environments. In [32], correct classification rate of 92% is reported when the testing environment is the same as the training environment, but this falls to 81.9% when the environments are different. While these results are not directly comparable with the results presented in this paper as these results are based on 20 s of speech and the corpora is different, they do suggest that previous methods struggle when noisy environments are used.

This is verified in our own experiments with identical training and test sets using a GMM method (based on the method proposed in [16]). Results are given (in Table V) for a training set comprising of *TIMIT* for the clean environment and *F1-TIMIT* for the noisy environments, the down-sampled and *F2-TIMIT* datasets are omitted from these experiments. Both PLP and MFCC features are used (as these have proved effective in previous work), a combination of these features is also used, these features are calculated in the same manner as described for the proposed GP system (only the mean values are considered). The number of clusters used is 32 (in common with previous work), and 15 iterations expectation maximization are used to train the system. While the cepstral mean subtraction (CMS) method has been used in some previous works, its advantages have not always been consistently observed [32]. For example, the above work shows that the average identification error of GMM using either PLP features or those with RASTA compensation

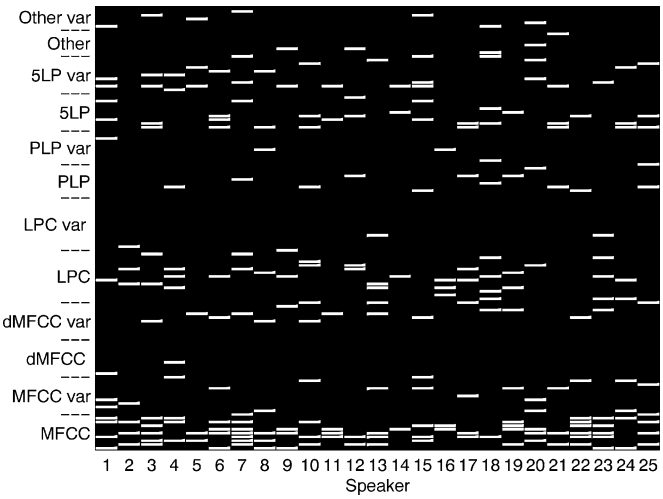

Fig. 6. Descriptors used in the solutions generated for the 25 TBV individuals under *normal* conditions. Note that descriptors marked *var* indicate the variance of that feature over the 1-s frame. All other values are *mean*. 5LP represents the five largest peaks (for other abbreviations see text).

in 1-s segments is not improved with the inclusion of CMS; in any case, CMS is not the issue here. As we aim to provide a baseline that can be computed accurately with the baseline in other work, we have not applied CMS in these investigations.

The results show that the GMM method is not as resilient to noisy transmission paths as the method we have proposed, this can be seen particularly when the training and testing environments are different. For completeness, it is worth noting that in our work GMM has 32 adjustable parameters, while GP has 36 functions and run parameters.

While the solutions generated by the GP process are very large, so clear understanding is difficult, some understanding can be gained by analysing which descriptors each solution uses. Fig. 6 shows the use of descriptors in solutions generated using the clean training set (TIMIT) and Fig. 7 shows the use of descriptors in solutions generated using the clean and noisy training set (TIMIT, F1-TIMIT, and F2-TIMIT-8). It is apparent that the solutions generated using the clean training set use less

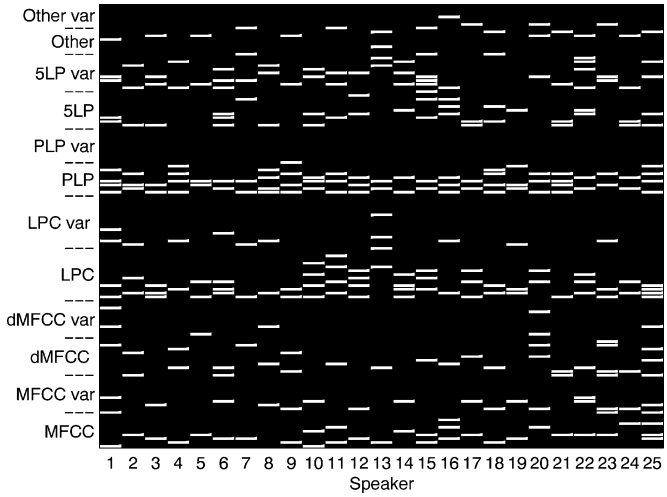

Fig. 7. Descriptors used in the solutions generated for the 25 TBV individuals under *noisy* conditions. See description of Fig. 6 for more details.

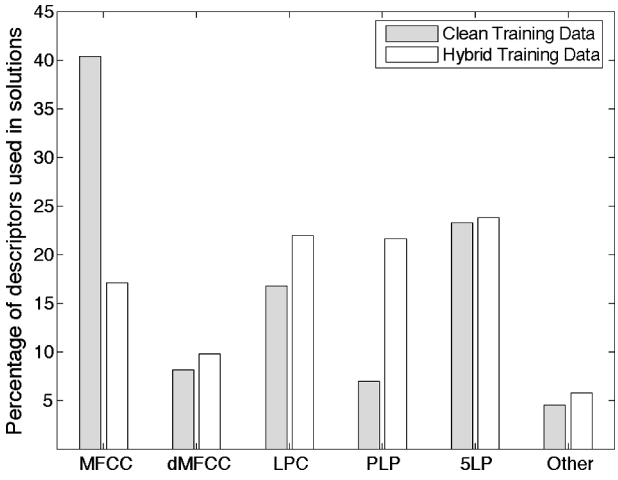

Fig. 8. Percentage of descriptors used for clean (TIMIT) and hybrid (TIMIT, F1-TIMIT, and F2-TIMIT-8) environments.

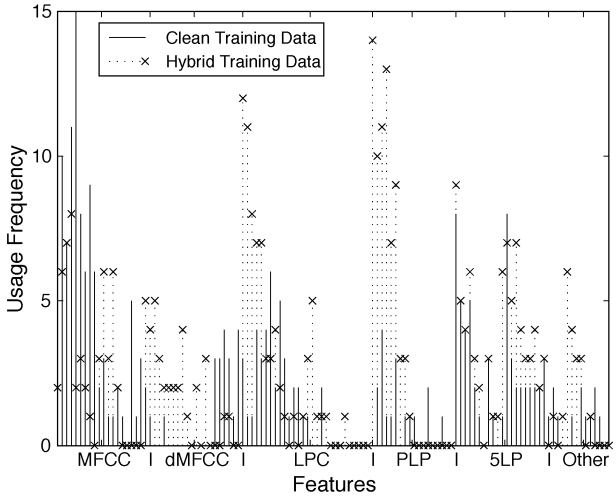

Fig. 9. Analysis of the frequency of features used for the clean (TIMIT) and hybrid (TIMIT, F1-TIMIT, and F2-TIMIT-8) environments.

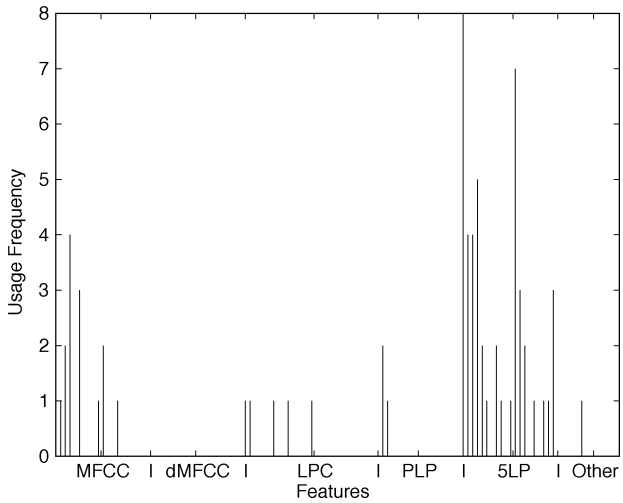

Fig. 10. An analysis of the frequency of features used for BOTH the clean (TIMIT) and hybrid (TIMIT, F1-TIMIT & F2-TIMIT-8) environments.

features (mean of 9.8) when compared to the solutions generated using the hybrid test set (mean of 13.2). It is also apparent that the features used are different for these two environments and that some features are more useful than others in ASV, in fact 21 features are not used by any of the solutions. The vast majority of the unused descriptors are *variance* measurements, indicating that the variance of the descriptors over a one second time frame is of less importance than the mean value over the same time frame. The total number of descriptors used in both environments is 80, and 61 of these are used in both environments indicating that these 61 descriptors both give an indication of the speakers identity and are somewhat resilient to noise.

Fig. 8 shows the percentage of each descriptor type used in each of these training environments. It can be seen that while the solutions generated using the clean training set are highly reliant on the MFCC (MFCCs accounting for around 40% of all descriptors used) the solutions generated using the hybrid test set are more reliant on PLP (PLPs account for 22% of all descriptors used). This is of particular interest because previous research has indicated that the use of PLPs gives better performance in noisy environments than MFCCs. (see [30]). A more fine-grain analysis can be seen in Fig. 9. These figures show that some of the five largest spectral peaks (5LP) give some useful information when performing ASV. While these descriptors alone may not indicate a speaker's identity, they may provide information useful in more complex solutions that consider the spectral peaks when analysing the more conventional spectral descriptors.

Fig. 10 shows the descriptor usage frequency for descriptors that are used in bot training environments for individual speakers. Interestingly there is not a great deal of overlap in the importance of specific descriptors in identifying specific speakers independent of environment. The exception to this is the descriptors based on the five largest spectral peaks (5LP), particularly the *magnitude* and *index variance* of the largest peak, which appear in eight and seven speaker-dependent solutions in both training environments, respectively.

The lack of significant overlap for other descriptors is most likely due to the redundancy present in these descriptor sets. The information about the 5 largest spectral peaks is independent of other spectral features which generally indicate the *shape* of the spectrum, and many solutions have found this unique information useful in evolving accurate classifiers. The other spectral descriptors are likely to have a much higher degree of overlap in the information they contain (i.e., they are less independent of one another). Consequently, different descriptors could be used to obtain the same speaker dependent information. The descriptor used is therefore likely to depend on the training environment. The results indicate that in a clean environment the MFCCs generally prove robust enough, while in the hybrid environment these appear to be replaced with PLPs.

V. Conclusion

GP shows great promise in creating robust classifiers for ASV purposes. Through the use of CPS, parallel GP (the island model) with dynamic migration paths and a dynamic fitness function, competitive results have been demonstrated. The generated programs can be evolved to be resilient to noisy transmission paths, this is believed to be largely due to the *per-voice* and *environment specific* feature selection inherent in the method. This method does not suffer the limitations associated with a limited feature set as a large number of feature can be used and only those useful are included in the final GP generated classification program.

It became apparent when analysing results that innovations of other researchers, such as the use of PLP in noisy environments, was naturally incorporated as part of the evolution process. Some of the descriptors were shown to be less important, and further work could exclude these in order to accelerate the evolutionary process. Equally, a more exhaustive descriptor set, perhaps including some high level features, could be used to increase performance further. Further work could also compare this system to a wider variety of ASV systems, and analyse the effect on performance of using other established techniques (such as CMS).

There is no reason why this classification method cannot be applied to other complex classification tasks with minimal adaptation and future applications are anticipated.

References

[1] J. P. Campbell, "Speaker recognition: a tutorial," *Proc. IEEE*, vol. 85, no. 9, pp. 1437–1462, Sep. 1997.

[2] T. Kinnunen, "Spectral Features for Automatic Text-Independent Speaker Recognition," Ph.D. dissertation, Univ. Joensuu, Joensuu, Finland, 2003.

[3] K. Gopalan, T. Anderson, and E. Cupples, "A comparison of speaker identification results using features based on cepstrum and fourier-bessel expansion," *IEEE Trans. Speech Audio Proces.*, vol. 7, no. 3, pp. 289–294, May 1999.

[4] D. Reynolds, W. Andrews, J. Campbell, J. Navratil, B. Peskin, A. Adam, Q. Jin, D. Klusacek, J. Abramson, R. Mihaescu, J. Godfrey, D. Jones, and B. Xiang, "The supersid project: Exploiting high level information for high-accuracy speaker recognition," in *Proc. ICASSP*, Hong Kong, 2003, pp. 784–787.

[5] L. Ferrer, H. Bratt, V. R. R. Gadde, S. Kajarekar, E. Shriberg, K. Sönmez, A. Stolcke, and A. Venkataraman, "Modeling duration patterns for speaker recognition," in *Proc. Eurospeech*, Geneva, Switzerland, 2003.

[6] H. Ezzaidi, J. Rouat, and D. O'Shaughnessy, "Towards combining pitch and MFCC for speaker identification systems," presented at the Eurospeech Conf., Aalborg, Denmark, 2001, Paper No. 2825, unpublished.

[7] G. Doddington, W. Liggett, A. Martin, M. Przybocki, and D. Reynolds, "Sheep, goats, lambs and wolves a statistical analysis of speaker performance in the nist 1998 speaker recognition evaluation," presented at the Int. Conf. Spoken Language Processing, Oct. 1998, Paper No. 608, unpublished.

[8] F. Bimbot, I. Magrin-Chagnolleau, and L. Mathan, "Second-order statistical measures for text-independent speaker identification," *Speech Commun.*, vol. 17, no. 1–2, pp. 177–192, Aug. 1995.

[9] J. Navratil, Q. Jin, W. Andrews, and J. Campbell, "Phonetic speaker recognition using maximum-likelihood binary-decision tree models," in *Proc. ICASSP*, Hong Kong, 2003.

[10] V. Wan, "Speaker Verification Using Support Vector Machines," Ph.D. disseration, Univ. Sheffield, Sheffield, U.K., Jun. 2003.

[11] V. Wan and W. M. Campbell, "Support vector machines for speaker verification and identification," in *Proc. Neural Networks for Signal Processing X*, 2000, pp. 775–784.

[12] R. Wouhaybi and M. A. Al-Alaou, "Comparison of neural networks for speaker recognition," in *Proc. 6th IEEE Int. Conf. Electronics, Circuits Systems (ICECS)*, Sep. 5–8, 1999, vol. 1, pp. 125–128.

[13] K. R. Farell, R. J. Mammone, and K. T. Assaleh, "Speaker recognition using neural networks and conventional classifiers," *IEEE Trans. Speech Audio Process.*, vol. 2, no. 1, pp. 194–205, Jan. 1994.

[14] R. P. Ramachandran, K. R. Farrell, R. Ramachandran, and R. J. Mammone, "Speaker recognition—general classifier approaches and data fusion methods," *Pattern Recognit.*, vol. 35, pp. 2801–2821, 2002.

[15] F. K. Soong, A. E. Rosenberg, L. R. Rabiner, and B. H. Huang, "A vector quantization approach to speaker recognition," *AT&T Tech. J.*, vol. 66, pp. 14–26, 1987.

[16] D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using gaussian mixture speaker models," *IEEE Trans. Speech Audio Process.*, vol. 3, no. 1, pp. 72–83, Jan. 1995.

[17] L. Rodriguez-Linares, C. Garcia-Mateo, and J. Alba-Castro, "On combining classifiers for speaker authentication," *Pattern Recognit.*, vol. 36, pp. 347–359, 2003.

[18] H. Guo, L. B. Jack, and A. K. Nandi, "Feature generation using genetic programming with application to fault classification," *IEEE Trans. Syst., Man, Cybern. B*, vol. 35, no. 1, pp. 89–99, Feb. 2005.

[19] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.

[20] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, *Genetic Programming—An Introduction*. New York: Morgan Kaufmann, 1998.

[21] T. Soule and J. A. Foster, "Effects of code growth and parsimony pressure on populations in genetic programming," *Evolut. Comput.*, vol. 6, no. 4, pp. 293–309, Winter 1998.

[22] S. Silva and J. Almeida, "Dynamic maximum tree depth," in Genetic and Evolutionary Computation-GECCO-2003. Chicago, IL, Jul. 12–16, 2003, vol. 2724, LNCS, pp. 1776–1787, Springer-Verlag.

[23] G. Folino, C. Pizzuti, G. Spezzano, V. Vanneschi, and M. Tomassini, "Diversity analysis in cellular and multipopulation genetic programming," Proc. 2003 Congr. Evolutionary Computation CEC2003 R. Sarker, R. Reynolds, H. Abbass, K. C. Tan, B. McKay, D. Essam, and T. Gedeon, Eds. Canberra, Dec. 8–12, 2003, pp. 305–311, IEEE Press.

[24] Koza, "Asynchronous "island" approach to parallelization of genetic programming," *Internet*, 2000.

[25] L. Lamel, R. Kassel, and S. Seneff, "Speech database development: design and analysis of the acoustic-phonetic corus," in *Proc. DARPA Speech Recognition Workshop*, 1986, pp. 100–110.

[26] W. Fisher, G. Doddington, and K. Goudie-Marshall, "The darpa speech recognition research database: specification and status," in *Proc. DARPA Speech Recognition Workshop*, 1986, pp. 93–100.

[27] D. Ellis, "Prediction-Driven Computational Auditory Scene Analysis," Ph.D. dissertation, Dept. Elect. Eng. Comput. Sci., Mass. Inst.Technol., Cambridge, MA, 1996.

[28] L. Lu, H. Jiang, and H. Zhang, "A robust audio classification and segmentation method," in *Proc. 9th ACM Int. Conf. Multimedia*, 2001, pp. 203–211.

[29] T. Zhang and C. Kuo, "Content-based classification and retrieval of audio," in *SPIE 43rd Annu. Meeting—Conf. Advanced Signal Processing Algorithms, Architectures, Implementations VII*, San Diego, CA, Jul. 1998.

[30] H. Hermansky, "Perceptual linear prediction (PLP) analysis for speech," *J. Acoust. Soc. Amer.*, vol. 87, pp. 1738–1752, 1990.

[31] J. R. Koza, M. A. Keane, S. Streeter, W. Mydlowec, J. Yu, and L. Lanza, *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*. Boston, MA: Kluwer, 2003.

[32] S. van Vuuren, "Comparison of text-independent speaker recognition methods on telephone speech with acoustic mismatch," in *Proc. ICSLP*, Oct 1996, pp. 1784–1787.

Peter Day received the B.Sc. (Hons.) degree (first class) in audio and music technology in 2001. He is currently pursuing the Ph.D. degree in the Signal Processing and Communications Group, Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, U.K.

His research interests include evolutionary computation, machine learning, and audio, speech, and music processing and analysis.

Asoke K. Nandi (SM'96) received the Ph.D. degree from the University of Cambridge (Trinity College), Cambridge, U.K., in 1979.

He held several research positions, including Rutherford Appleton Laboratory (U.K.), the European Organisation for Nuclear Research (Switzerland), the Department of Physics, Queen Mary College, London, U.K., and the Department of Nuclear Physics, Oxford, U.K. In 1987, he joined Imperial College, London, as the Solartron Lecturer in the Signal Processing Section of the Electrical Engineering Department. In 1991, he joined the Signal Processing Division of the Electronic and Electrical Engineering Department, University of Strathclyde, Glasgow, U.K., as a Senior Lecturer; subsequently, he was appointed a Reader in 1995 and a Professor in 1998. In March 1999, he moved to the University of Liverpool, Liverpool, U.K., to take up his appointment to the David Jardine Chair of Signal Processing in the Department of Electrical Engineering and Electronics. In 1983, he was a member of the UA1 team at CERN that discovered the three fundamental particles known as W^+, W^- and Z^o providing the evidence for the unification of the electromagnetic and weak forces, which was recognized by the Nobel Committee for Physics in 1984. Currently, he is the Head of the Signal Processing and Communications Research Group with interests in the areas of non-Gaussian signal processing, communications and machine learning research. With his group, he has been carrying out research in machine condition monitoring, signal modelling, communication signal processing, biomedical signals, development and applications of machine learning, blind source separation, and blind deconvolution. He has authored or co-authored over 300 technical publications, including two books: *Automatic Modulation Recognition of Communications Signals* (Boston, MA: Kluwer Academic, 1996) and *Blind Estimation Using Higher-Order Statistics* (Boston, MA: Kluwer Academic, 1999).

Dr. Nandi was awarded the Mounbatten Premium, Division Award of the Electronics and Communications Division, of the Institution of Electrical Engineers of the U.K. in 1998 and the Water Arbitration Prize of the Institution of Mechanical Engineers of the U.K. in 1999. He is a Fellow of the Cambridge Philosophical Society, the Institution of Electrical Engineers, the Institute of Mathematics and its applications, the Institute of Physics, and the Royal Society for Arts.