Speaker Verification on Unbalanced Data with Genetic Programming

Róisín Loughran[1(✉)], Alexandros Agapitos[1], Ahmed Kattan[2], Anthony Brabazon[1], and Michael O'Neill[1]

[1] Natural Computing Research and Applications Group, University College Dublin, Dublin, Ireland
{roisin.loughran,alexandros.agapitos}@ucd.ie
[2] Computer Science Department, Um Al-Qura University, Mecca, Saudi Arabia

Abstract. Automatic Speaker Verification (ASV) is a highly unbalanced binary classification problem, in which any given speaker must be verified against everyone else. We apply Genetic programming (GP) to this problem with the aim of both prediction and inference. We examine the generalisation of evolved programs using a variety of fitness functions and data sampling techniques found in the literature. A significant difference between train and test performance, which can indicate overfitting, is found in the evolutionary runs of all to-be-verified speakers. Nevertheless, in all speakers, the best test performance attained is always superior than just merely predicting the majority class. We examine which features are used in good-generalising individuals. The findings can inform future applications of GP or other machine learning techniques to ASV about the suitability of feature-extraction techniques.

Keywords: Speaker verification · Unbalanced data · Genetic programming · Feature selection

1 Introduction

Automatic Speaker Verification (ASV) is the process of accurately verifying that a speaker is who they claim to be. This is feasible because each individual's voice is audibly unique due to physical attributes such as length of vocal tract, size of larynx etc. along with habitual characteristics such as accent and inflection. ASV has important applications in the fields of phone banking, shopping and security systems.

ASV is inherently a highly unbalanced binary-classification problem since it requires to accurately recognise one speaker from everyone else; the first class A contains examples from to-be-verified speaker, whereas the second class B contains examples from the rest of the speakers (i.e. impostors). Class A is the *minority class* as it is often represented with a smaller number of training examples, while class B is the *majority class*. This imbalance in class distribution is a significant problem; it introduces a learning bias and often results in classification models that are not accurate in the cases of the to-be-verified speaker.

G. Squillero and P. Burelli (Eds.): EvoApplications 2016, Part I, LNCS 9597, pp. 737–753, 2016.
DOI: 10.1007/978-3-319-31204-0_47

In general, in class imbalance problems, the smaller the ratio of minority class examples to majority class examples, then the stronger this bias becomes and the harder it is for a classifier to generalise [4].

With the notable exception of [9], ASV is an application area that has received little attention from the GP community. This paper reports a preliminary empirical study that approaches the problem of ASV from the class imbalance perspective. A number of different fitness functions and training-data sampling techniques found in the literature are examined for their efficiency to evolve good-generalising programs. The simulations herein are performed on the TIMIT corpora [13], a regularly-used dataset for speaker recognition and verification.

Section 2 provides background information on ASV, and on methods for tackling class imbalance issues in pattern classification algorithms in general and in GP in particular. Section 3 details the scope of the experiments. Section 4 introduces the TIMIT corpora, presents methods for feature extraction, details the GP systems under comparison and the setup of the experiments. Section 5 analyses the empirical results, while Sect. 6 concludes and discusses future work.

2 Background

The aim of this section is to first provide an overview of traditional classification models for ASV. It then briefly describes the two main categories of methods for tackling class imbalance problems. The final part reviews GP work based on unbalance datasets.

2.1 Automatic Speaker Verification

Early speaker verification models were based on Vector Quantisations [6]. Another prominent classification method for ASV to emerge in the early 1990s was based on Gaussian Mixture models [31]. Over the next three decades other classification techniques were applied to the problem of speaker recognition and verification, such as Support Vector Machines (SVM) [7], Artificial Neural Networks [32], ensemble learning [25], and Genetic Programming [9].

A number of recent studies have focussed on methods to counteract inter-speaker and inter-session variability by examining channel compensation between recordings. Such studies have used feature mapping to transform obtained features into a channel-independent feature-space. Methods such as Joint Factor Analysis [20], i-vectors [21] and PDLA [10] were used for this purpose. This focus on channel effects is in part driven by the NIST Speaker Recognition Evaluation[1] which evaluates novel speaker recognition systems on a corpora of phone recordings.

[1] http://www.nist.gov/itl/iad/mig/ivec.cfm.

2.2 Tackling Class Imbalance

There exist a number of methods for learning good-generalising classifiers for class imbalance datasets. The taxonomy of these methods mainly consists of two major categories; those of *training data sampling* and *cost-sensitive training*. The work of [4] provides an excellent overview of these methods, with references from both statistical machine learning and GP. For the sake of completeness we very briefly introduce the two dominant classes in the following sections.

Training-Data Sampling. Balancing of training examples can be achieved either by *over-sampling* the minority class or *under-sampling* the majority class [2]. *Synthetic over-sampling* and *editing* have been often shown to be superior to the sampling techniques described above. Synthetic oversampling of the minority class creates additional examples by interpolating between several similar examples [3], while editing removes noisy or atypical examples from the majority class [23].

Cost-Sensitive Training. In a classification problem, we are given a training set of N examples $\{(x_i, y_i)\}_{i=1}^{N}$, where $x \in \mathbb{R}^d$ is a d-dimensional vector of explanatory variables and $y \in C = \{1, \ldots, c\}$ is a discrete response variable, with joint distribution $P(x, y)$. We seek a function $f(x)$ for predicting y given the values of x. The loss function $L(y, f(x))$ for penalising errors in prediction can be represented by a $K \times K$ cost matrix L, where $K = card(C)$. L will be zero on the diagonal and non-negative elsewhere, where $L(k, l)$ is the price paid for misclassifying an observation belonging to class C_k as C_l. Most often, in cases of balanced datasets, a *zero-one* loss function $L(y, f(x)) = I(y \neq f(x))$[2] is used, where all misclassifications are charged one unit. In the case of unbalanced datasets, the cost matrix can be adjusted to increase the cost of misclassifying the examples of the minority class.

2.3 GP on Unbalanced Datasets

Genetic Programming has been applied to unbalanced datasets in a number of studies. Work using the data sampling techniques of Random Sampling Selection (RSS) and Dynamic Subset Selection (DSS) is reported in [8,14]. In [8] a two-level sampling approach is first used to sample blocks of training examples using RSS and then select examples from within those blocks using DSS. In [14] DSS is used to bias the selection of training examples towards hard-to-classify examples, while RSS was used to bias towards the selection of minority class training examples.

Cost adjustment strategies usually focus on adapting the fitness function to reward programs which have good accuracy on both classes with better fitness, while penalising those with poor accuracy on one class with low fitness. The use

[2] $I(\cdot)$ is the indicator function.

of different misclassification costs to incorrect class predictions is reported in [18]. In the work of [12] an adaptive fitness function increases misclassification costs for difficult-to-classify examples. In [33] RSS and DSS are used in conjunction with three novel fitness functions with an application to a network intrusion detection problem. The work of [29] used both rebalancing of data and cost-sensitive fitness functions in comparing GP with other data-mining approaches to predict the rate of student failure in school. The work of [4] used six data sets with different class imbalance ratios and applied GP with a number of different fitness functions. A multi-objective GP approach for evolving accurate and diverse ensembles of GP classifiers that perform well on both minority and majority classes was proposed in [5]. A weighted average composed of error rate, mean squared error and a novel measure of class separability similar to Area Under Curve is used in [34]. In the work of [11], data sub-sampling is used in combination with the average of the geometric mean between minority and majority class accuracies and the Wilcoxon-Mann-Whitney statistic.

3 Scope of Research

To the best of our knowledge, the application of GP to speaker verification has only been reported in the work of [9]. One of the principal applications of ASV systems is remotely confirming the identity of a person for reasons of security such as telephone banking. The literature review conducted in [9] showed that while good results have been reported using a variety of different systems of statistical machine learning on noiseless input signals, most systems suffer heavily if the signal is transmitted over a noisy transmission path (i.e. a telephone network). In order to create a "noisy" environment, several datasets were derived from the original TIMIT corpora using filters that included both additive and convolutive noise. GP experiments were then set to evolve classifiers based on extracted features impaired by noise. Twenty-five speakers to-be-verified and forty-five "impostors" were selected from the TIMIT corpora. For each of the to-be-verified speakers the training set consists of fifteen seconds of to-be-verified speech and forty-five seconds of impostor speech (one second of randomly selected speech from each of the forty-five impostor individuals). This results in a minority class to majority class imbalance ratio of 1:3. A pool of hand-engineered features were extracted from the raw signal and populated the terminal set. The fitness function used was dynamically biased to concentrate on the most difficult-to-classify examples. Finally, an island model was employed to improve population diversity. Results showed that generated programs can be evolved to be resilient to noisy transition paths, which was mainly attributed to the speaker-dependent and environment-specific feature selection inherent in GP.

In this paper we investigate a different facet of the GP application to ASV. The research scope is two-fold. First, we study the generalisation of GP-evolved programs on ASV datasets that exhibit a high class imbalance ratio. Specifically, the experiments designed are based on datasets with a class imbalance ratio

of 1:9. We used the original, noiseless TIMIT corpora. A number of different methods for cost-sensitive training and data sampling are compared in terms of their effectiveness to assist with the evolution of good-generalising programs.

The second aim of this paper is to inform future applications of GP and other machine learning algorithms to ASV in terms of the usefulness of different features for constructing classifiers. For this purpose, we extract 275 features from the raw signal to create program input, and rely on the inherent ability of GP to perform feature selection. We analyse the terminal-nodes of highly-performing programs, and calculate statistics on the frequency of usage of different features.

4 Methods

4.1 Speaker Corpus

The speech recordings used in this study are taken from the TIMIT corpora [13]. This was chosen due its very regular use in the speaker recognition and verification literature. The corpora consists of 630 speakers, 192 female and 438 male, from 8 American dialects each reading 10 phonetically rich sentences. Each sentence was recorded on a high quality microphone at a sampling rate of 16 kHz.

4.2 Training and Test Data

For these experiments we chose a random 10 speakers, 4 female and 6 male, from the corpus and developed a classifier for each speaker. For each experiment the audio from the given speaker is the to-be-verified minority class and the audio from the nine other speakers constitute the majority class. In this manner we create a 1:9 class imbalance ratio for each experiment.

Each speaker offers 10 utterances of approximately 3 s each. To increase the number of speech utterances, we split each sentence into three equal parts of approximately 1 s. Early analysis showed that the third part of each sentence was of lower timbral quality than the preceding sections. Thus only the first two thirds of each sentence were included in the learning dataset of 200 examples. In the experiments, a *training set* of size 120 examples is used to evolve programs, and a *test set* of 80 examples is used to assess generalisation.

The features calculated on this data are detailed in Sect. 4.4. Rather than reducing these features using the statistical mean or variance of the windowed signal, we employed Principal Component Analysis (PCA) on a number of the high-dimensional features. PCA was used on these results to record the maximum variance within each feature while reducing the dimensionality of the data. In total this resulted in 275 features calculated on 200 data samples.

4.3 GP Systems

A number of systems tailored to unbalanced classification problems from the literature were chosen for this study. These are detailed below.

ST. This system is trained using the original unbalanced dataset. We employ a version of the MSE-based loss function that has been shown [4] to improve upon the performance of fitness functions based on classification accuracy[3] or the weighted average of true positive and true negative rates. Given N training examples $\{(x_i, t_i)\}_{i=1}^{N}$ containing the examples of both majority and minority classes, L_{MSE} is defined as:

$$L_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\Phi(f(x_i)) - t_i)^2 \tag{1}$$

where

$$\Phi(x) = \frac{2}{1+e^{-x}} - 1 \tag{2}$$

and $f(x_i)$, t_i is the program output and target value for the i^{th} training case respectively. The sigmoid function in Eq. 2 scales $f(x)$ within the range of $\{-1, \ldots, 1\}$. Similarly to [4], the target value for the majority class is set to -0.5, while the target value for the minority class is set 0.5. Classification is based on a zero-threshold approach; positive program output is mapped to the minority class label, while negative output is mapped to the majority class label.

AVE. This system is trained using the original unbalanced dataset. The loss function uses a weighted-average classification accuracy of the minority and majority classes [4]. Minority accuracy corresponds to the true positive rate, whereas majority accuracy corresponds to true negative rate. The weighting coefficient between the two is $0 < w < 1$. When w is set to 0.5, the accuracy of both classes contributes equally to the loss function. In case of $w > 0.5$ the accuracy of the minority class contributes more to the loss function, lowering the contribution of the majority class accuracy. The loss function L_{AVE} is defined as:

$$L_{AVE} = 1.0 - \left(w \times \frac{TP}{TP+FN} + (1-w) \times \frac{TN}{TN+FP}\right) \tag{3}$$

where TP, TN, FN, FP is the count of true positives, true negatives, false negatives and false positives respectively.

US. Since static under-sampling of the majority class examples can introduce unwanted sampling bias and discard potentially useful training examples, we resort to a dynamic version of under-sampling. At every generation, a new set of examples is drawn random-uniformly from the set of training examples of the majority class. Under-sampling ensures that the number of examples drawn from the majority class is the same as the number of examples for the minority class. The loss function used is given in Eq. 1.

[3] The number of examples correctly classified as a fraction of the total number of training examples.

RS. A type of random sampling technique in which programs are evaluated on a *single* example drawn uniform-randomly from the entire training dataset in each generation was shown to improve the generalisation of programs as compared to the use of the complete training set [15]. An obvious extension of this method to datasets with class imbalance is to populate the training set with *two* randomly-drawn examples (different in each generation), one each from the minority and majority class. The loss function used is given in Eq. 1.

4.4 Feature Extraction

High-level features describing fundamental frequency or rhythm are difficult to measure accurately, possible to mimic and susceptible to emotions. Thus lower level spectral, cepstral, spectro-temporal and statistical features are more common for speaker verification. A survey of the literature indicates that the following short-term spectral features are the recommended features to include [22].

Mel-frequency Cepstral Coefficients. MFCCs have become the standard measure of speech analysis for some time [30]. They consist of a set of coefficients that can represent the spectral quality within a sound according to a scale based on human hearing. Obtaining the MFCCs consists of windowing the sound, calculating amplitude spectrum of cepstral feature vector for each frame and then converting this to the perceptually derived *mel-scale* [26]. The dimensionally of the MFCCs were reduced in this study using PCA. The first four PCs of the first 12 MFCCs along with their derivatives are included in these experiments.

Linear Prediction Coefficients. Linear prediction calculates a given signal based on a linear combination of the previous inputs and outputs [28]. As a spectrum estimation it offers good interpretation in both the time and frequency domains. In the time domain, LP predicts according to

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

Where $s[\tilde{n}]$ is the predicted signal, s[n] is the observed signal and a_k are the predictor coefficients. The prediction error or residual is defined as the difference between the predicted signal and the observed signal:

$$e[n] = s[n] - s[\tilde{n}] \tag{5}$$

The linear predictive coefficients (LPCs), a[k], are determined by minimising this residual. This analysis leads to the *Yule-walker equations* that can be efficiently solved using *Levinson-Durbin recursion* [19]. Given the LPC coefficients *a[k], k = 1, . . . ,p*, the linear predictive cepstral coefficients (LPCCs) are computed using the recursions:

$$c[n] = \begin{cases} a[n] + \sum_{k=1}^{n-1} \frac{k}{n} c[k] a[n-k] & \text{if } 1 \le n \le p \\ \sum_{k=n-p}^{n-1} \frac{k}{n} c[k] a[n-k] & \text{if } n > p. \end{cases} \tag{6}$$

Table 1. Function/Terminal sets and run parameters

PRIMITIVE LANGUAGE	
Function set	+, −, ∗, / (x/y returns x if $\|y\| < 10^{-5}$), *sin*, *cos*, e^x, *log* (log(x) returns x if $x \leq 0$), *sqrt* (sqrt(x) returns x if $x < 0$)
Terminal set	275 features 40 uniform-randomly drawn constants in the range of $[-1.0, 1.0]$
GP PARAMETERS	
Evolutionary algorithm	elitist (1% of population size), generational
Population size	1,000
Tournament size	4
No. of generations	51 for ST 51 for AVE 251 for US 3,001 for RS
Population initialisation	ramped half-and-half (depths of 2 to 4)
Max. tree depth	8

The first 21 LPCs and 10 LPCCs (apart from the zeroth order) were included in our dataset. An equivalent measure to these that has become popular in speaker analysis is line spectral frequencies (LSFs) [19]. These can be useful in practice as they result in low spectral distortion and are deemed to be more sensitive and efficient than other equivalent representations.

Perceptual Linear Prediction. One downfall of the LP method is that it approximates the spectrum of speech equally well at all frequencies. In contrast, after 800 Hz, the human ear becomes less sensitive and spectral resolution decreases with frequency. This is compensated for by using Perceptual Linear Prediction (PLP) [16]. The RelAtive SpecTrAl (RASTA) [17] method was developed to make PLP more robust to linear spectral distortions by replacing the short-term spectrum by a spectral estimate. This suppresses any slow varying component making the spectral estimate of that channel less sensitive to slow variations.

Other Features. A number of descriptive spectral features were also included in our dataset. These included the Spectral Centroid, Inharmonicity, Number of Spectral Peaks, Zero Crossing Rate, Spectral Rolloff, Brightness, Spectral Regularity and the Spectral Spread, Skewness and Kurtosis. Many of these were calculated using the MIRToolbox [24], a Matlab toolbox dedicated to the extraction of musically-related features from audio recordings.

4.5 Primitive Language, Variation Operators, GP Parameters

The primitive language and the evolutionary run parameters are given in Table 1. Over 50 generations, the ST and AVE methods perform 6,000 fitness evaluations. To ensure a fair comparison the number of generations used in the US and RS methods are adjusted accordingly. Preliminary experiments revealed a tendency of all systems to overfit, thus the maximum tree-depth is set to 8 to restrict the complexity of the evolved programs.

The search strategy that we employed relies heavily on mutation-based variation operators. The operation of `pointMutation(x)` traverses the tree

in a depth-first manner, and depending on the probability x it substitutes a tree-node by another random tree-node of the same arity. The operation of `subtreeMutation()` selects a node uniform-randomly and replaces the subtree rooted at that node with a newly generated subtree. The tree-generation procedure is *grow* or *full*, each applied with equal probability. To improve on the exploratory effect of the mutation operator, other than picking the tree-node to be replaced from the whole expression-tree, we devised an additional node-selection method. In this method a depth-level is picked uniform-randomly from the range of all possible depth-levels present in the expression-tree, and subsequently a node is picked uniform-randomly from the set of nodes that lie in the chosen depth-level. The decision between the two node-selection methods is governed by a probability set to 0.5 for both methods. Finally, our implementation of recombination operator is the standard subtree crossover defined for expression-tree representations. The probability of selecting an inner-node as a crossover point is set to 0.9, while the probability of selecting a leaf-node is set to 0.1.

In generating offspring, a probability is associated with applying either mutation of crossover, set to 0.7 in favour of mutation. If mutation is chosen, `pointMutation(0.1)` is applied with a probability of 0.1, `pointMutation(0.2)` is applied with a probability of 0.1, `pointMutation(2 / tree_size)` is applied with a probability of 0.2, and `subtreeMutation()` is applied with a probability of 0.6.

5 Results

We created 50 splits of the 200 learning examples into training and test sets. In each split, 120 examples are drawn uniform-randomly for the training set, while the remaining 80 examples populate the test set. Stratification ensures that the class imbalance ratio is maintained in both sets. Using each split, we performed 50 independent evolutionary runs using each GP system. Many practitioners use an equal weighing in the AVE system by setting $w = 0.5$ [4]. In this work the effectiveness of AVE is evaluated using a set of values for w, that of $W = \{0.5, 0.6, 0.7, 0.8\}$. In the experiments we performed no model selection, thus the fittest individual (on the training dataset) of the last generation is designated as the output of a run.

5.1 Generalisation Performance

Table 2 presents statistics of training and test *classification accuracy* for the different systems on all 10 speakers. In each case, we report the median, interquartile range, and maximum based on 50 independent runs. Note that in a classification setup, in which the true positive rate corresponds to the minority class accuracy, a classifier that always outputs the majority class label attains a classification accuracy of 0.9 (true positive rate of 0 %). Our first observation concerns the significant difference between training and test performance in all datasets.

Table 2. Performance summary. Interquartile range in parentheses.

TRAINING CLASSIFICATION ACCURACY								
	AVE($w = 0.5$)		ST		RS		US	
Speaker id	Median	Max	Median	Max	Median	Max	Median	Max
FGRW0	0.97 (0.04)	1.00	0.98 (0.09)	1.00	0.96 (0.03)	0.99	0.99 (0.03)	1.00
FJEN0	0.97 (0.05)	1.00	0.95 (0.09)	1.00	0.97 (0.02)	0.98	0.98 (0.03)	1.00
FPJF0	0.99 (0.03)	1.00	0.93 (0.08)	1.00	0.97 (0.03)	1.00	0.98 (0.03)	1.00
FSAH0	0.97 (0.05)	1.00	0.90 (0.06)	0.98	0.97 (0.03)	0.99	0.98 (0.03)	1.00
MEFG0	1.00 (0.02)	1.00	0.98 (0.07)	1.00	0.98 (0.02)	1.00	0.99 (0.01)	1.00
MJDC0	0.97 (0.04)	1.00	0.97 (0.07)	1.00	0.96 (0.01)	0.98	0.99 (0.03)	1.00
MKDD0	0.98 (0.04)	1.00	0.97 (0.09)	1.00	0.97 (0.03)	1.00	0.99 (0.02)	1.00
MMGC0	0.95 (0.04)	1.00	0.92 (0.07)	0.99	0.96 (0.03)	0.98	0.97 (0.03)	1.00
MPGR1	0.97 (0.03)	1.00	0.97 (0.06)	1.00	0.96 (0.03)	0.98	0.98 (0.04)	1.00
MTRT0	0.96 (0.03)	1.00	0.90 (0.05)	0.98	0.96 (0.03)	0.99	0.97 (0.03)	1.00
TEST CLASSIFICATION ACCURACY								
	AVE($w = 0.5$)		ST		RS		US	
Speaker id	Median	Max	Median	Max	Median	Max	Median	Max
FGRW0	0.90 (0.06)	0.97	0.91 (0.03)	0.96	0.90 (0.04)	0.97	0.93 (0.04)	0.97
FJEN0	0.85 (0.06)	0.91	0.91 (0.01)	0.95	0.89 (0.03)	0.96	0.88 (0.02)	0.96
FPJF0	0.86 (0.05)	0.95	0.90 (0.04)	0.95	0.88 (0.04)	0.94	0.89 (0.10)	0.96
FSAH0	0.84 (0.05)	0.91	0.90 (0.01)	0.95	0.89 (0.04)	0.96	0.90 (0.05)	0.95
MEFG0	0.93 (0.05)	0.99	0.90 (0.05)	0.99	0.94 (0.03)	0.97	0.91 (0.05)	0.96
MJDC0	0.90 (0.08)	0.96	0.90 (0.04)	0.96	0.90 (0.04)	0.94	0.88 (0.02)	0.94
MKDD0	0.93 (0.07)	1.00	0.90 (0.04)	0.95	0.91 (0.05)	0.97	0.94 (0.04)	1.00
MMGC0	0.81 (0.14)	0.94	0.91 (0.01)	0.94	0.86 (0.04)	0.93	0.86 (0.06)	0.94
MPGR1	0.79 (0.10)	0.94	0.92 (0.03)	0.93	0.88 (0.06)	0.94	0.85 (0.05)	0.93
MTRT0	0.89 (0.07)	0.94	0.90 (0.01)	0.94	0.88 (0.05)	0.94	0.90 (0.05)	0.95

This is indicative of overfitting, a typical problem in unregularised GP [1]. There are a number of reasons why overfitting is occurring in these preliminary experiments. First and foremost, this is attributed to the limited number of examples for the to-be-verified speakers in each dataset. A second reason is the absence of both model selection and regularisation from the learning process. In light of the above, we attempted to limit the syntactic complexity of the evolved programs by setting the maximum tree-depth allowed during search to 8, however this was not adequate for preventing overfitting.

The generalisation performance of different systems is presented in the second part of Table 2. Table 4 presents the p-values of a two-sided Wilcoxon rank sum test, which tests the *null* hypothesis that two data samples have equal medians, against the alternative that they don't. We set the significance level α to 0.05. Median test accuracy of ST is shown to be statistically superior to rest of the systems AVE, RS, US for speakers FJEN0, MMGC0, MPGR1. In addition, ST median is shown to be statistically superior against that of (a) AVE for speakers FPJF0, FSAH0; (b) RS for speakers FGRW0, FPJF0, MTRT0; and (c) US for speakers MJDC0. This result is consistent with the findings in [4], which showed that the MSE-based loss function of Eq. 1 routinely outperformed loss functions based on classification accuracy or the weighted average between true positive and true negative rates (Eq. 3). The results also suggest that ST, which uses the original unbalanced datasets, is often statistically superior or no different to the data-sampling methods of RS and US. Specifically ST is statistically better in 6/10 speakers, and statistically worse in 1/10 speakers against RS. Also, ST

Table 3. Test classification accuracy for AVE. Interquartile range in parentheses.

	AVE($w = 0.5$)		AVE($w = 0.6$)		AVE($w = 0.7$)		AVE($w = 0.8$)	
Speaker id	Median	Max	Median	Max	Median	Max	Median	Max
FGRW0	0.90 (0.06)	0.97	0.88 (0.11)	0.99	0.84 (0.10)	0.97	0.86 (0.14)	0.96
FJEN0	0.85 (0.06)	0.91	0.85 (0.06)	0.93	0.82 (0.10)	0.90	0.82 (0.10)	0.93
FPJF0	0.86 (0.05)	0.95	0.89 (0.12)	0.95	0.86 (0.09)	0.95	0.86 (0.06)	0.94
FSAH0	0.84 (0.05)	0.91	0.86 (0.06)	0.94	0.82 (0.10)	0.96	0.84 (0.06)	0.91
MEFG0	0.93 (0.05)	0.99	0.93 (0.06)	1.00	0.91 (0.06)	0.95	0.94 (0.05)	0.97
MJDC0	0.90 (0.08)	0.96	0.86 (0.06)	0.96	0.78 (0.12)	0.90	0.84 (0.08)	0.95
MKDD0	0.93 (0.07)	1.00	0.88 (0.12)	0.97	0.91 (0.10)	0.99	0.88 (0.12)	0.99
MMGC0	0.81 (0.14)	0.94	0.81 (0.08)	0.90	0.81 (0.09)	0.90	0.84 (0.07)	0.94
MPGR1	0.79 (0.10)	0.94	0.79 (0.06)	0.93	0.85 (0.09)	0.93	0.82 (0.10)	0.94
MTRT0	0.89 (0.07)	0.94	0.85 (0.10)	0.94	0.85 (0.07)	0.96	0.84 (0.12)	0.95

is statistically better in 4/10 speakers, and statistically worse in 1/10 speakers against US.

Overall, the median of ST is equal to 90 % in 6/10 speakers, and greater than 90 % in 4/10 speakers. This suggests that in 50 runs, the median generalisation performance of ST is consistently equal or better to the performance of a classifier that always outputs the majority class label. The median test accuracy is higher than or equal to 90 % in 4/10 speakers for AVE; 4/10 speakers for RS; and 5/10 speakers for US. Nevertheless, among 50 runs, the maximum test classification accuracy that is achieved through evolution is always higher than the one yielded from the classifier that always outputs the majority class label, for all speakers.

Table 3 presents the test accuracy statistics for the different values of w in the loss function (Eq. 3) of the Ave system. A two-sided Wilcoxon rank sum test is performed to test the difference in the median values. The table with

Table 4. p-values of Winlcoxon rank-sum test. AVE uses $w = 0.5$.

		ST	RS	US
FGRW0	AVE	0.27	0.50	0.03
	ST		0.03	0.08
	RS			0.00
FJEN0	AVE	0.00	0.00	0.00
	ST		0.04	0.00
	RS			0.09
FPJF0	AVE	0.02	0.32	0.20
	ST		0.02	0.92
	RS			0.23
FSAH0	AVE	0.00	0.00	0.00
	ST		0.17	0.47
	RS			0.16
MEFG0	AVE	0.00	0.42	0.01
	ST		0.00	0.65
	RS			0.00
MJDC0	AVE	0.17	0.79	0.05
	ST		0.03	0.00
	RS			0.00
MKDD0	AVE	0.35	0.71	0.00
	ST		0.12	0.00
	RS			0.00
MMGC0	AVE	0.00	0.00	0.00
	ST		0.00	0.00
	RS			0.87
MPGR1	AVE	0.00	0.00	0.00
	ST		0.01	0.00
	RS			0.00
MTRT0	AVE	0.21	0.28	0.19
	ST		0.00	0.80
	RS			0.00

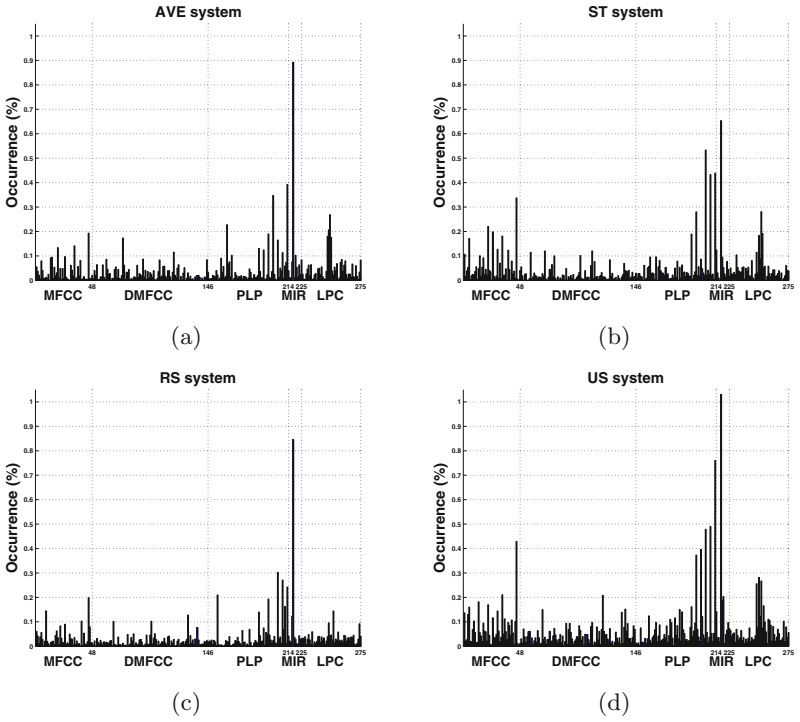

Fig. 1. Mean of the occurrence of all features in the top performing programs from independent runs for the AVE, ST, RS and US systems described in Sect. 4.3. Features are grouped into the mel-frequency cepstral coefficients (MFCC), their derivatives (DMFCC), perceptual linear prediction (PLP), other spectral features (MIR) and the linear prediction coefficients (LPC).

the p-values is omitted due to space limitations. We found that no value of w, where $w \neq 0.5$ shows a significantly better test classification accuracy compared to equal weighing. This finding is in accordance with the result reported in [4].

5.2 Feature Selection

We can determine the most beneficial features for the given problem by examining which features are most often chosen by high performing trees. A similar method has been used for feature selection in musical instrument analysis [27]. In examining these successful features, we only considered those programs from 50 independent GP runs that attained a classification accuracy greater than 90 %. A plot of the mean percentage of times each feature is chosen by a successful classifier is shown in Fig. 1. A more detailed account of the top 20 chosen features for each system is given in Table 5. This table names each of the top chosen features, reporting the mean percentage of times this features was chosen from the list of all 275 features and the standard error of this selection.

Table 5. Top 20 features for each system. Values are reported as the mean percentage of times each feature was chosen by a successful classifier (accuracy greater than 94 %) with the standard error in parenthesis.

Ave		ST		RS		US	
Feature	Mean(%)	Feature	Mean(%)	Feature	Mean(%)	Feature	Mean(%)
Inharm	0.89 (0.38)	Inharm	0.66 (0.2)	Inharm	0.85 (0.16)	Inharm	1.03 (0.23)
plp9_1	0.39 (0.18)	plp7_1	0.53 (0.1)	plp7_1	0.3 (0.11)	plp9_1	0.76 (0.15)
plp6_1	0.34 (0.27)	plp9_1	0.44 (0.11)	plp8_1	0.27 (0.06)	plp8_1	0.49 (0.18)
lpcc4	0.27 (0.1)	plp8_1	0.43 (0.11)	plp9_1	0.24 (0.06)	plp7_1	0.47 (0.09)
plpRast5_2	0.23 (0.13)	mfcc12_1	0.34 (0.08)	plpRast3_2	0.21 (0.01)	mfcc12_1	0.43 (0.1)
lpcc3	0.2 (0.15)	lpcc7	0.28 (0.09)	mfcc12_1	0.19 (0.03)	plp6_1	0.4 (0.16)
mfcc12_1	0.19 (0.04)	plp5_1	0.28 (0.1)	plp5_1	0.19 (0.05)	plp5_1	0.37 (0.1)
plp5_1	0.19 (0.1)	mfcc6_1	0.22 (0.1)	plp8_3	0.16 (0.05)	lpcc5	0.28 (0.14)
lpcc2	0.18 (0.17)	mfcc7_1	0.2 (0.08)	mfcc3_1	0.14 (0.04)	lpcc7	0.27 (0.16)
lpcc5	0.18 (0.1)	lpcc8	0.19 (0.09)	lpcc7	0.14 (0.03)	lpcc3	0.26 (0.1)
Dmfcc7_2	0.17 (0.07)	plp4_1	0.19 (0.07)	plp3_1	0.14 (0.09)	mfcc9_1	0.21 (0.06)
plp7_1	0.17 (0.06)	lpcc5	0.18 (0.09)	DDmfcc9_1	0.13 (0.02)	DDmfcc6_2	0.21 (0.06)
mfcc9_1	0.14 (0.07)	mfcc9_1	0.18 (0.06)	Centroid	0.12 (0.02)	ZeroCross	0.2 (0.09)
mfcc5_3	0.14 (0.09)	mfcc2_1	0.17 (0.08)	mfcc10_3	0.1 (0.00)	NoPeaks	0.19 (0.1)
plp3_1	0.13 (0.12)	mfcc8_1	0.12 (0.06)	Dmfcc5_2	0.1 (0.00)	mfcc4_1	0.18 (0.05)
plp4_1	0.12 (0.08)	plp9_2	0.12 (0.05)	DDmfcc1_2	0.1 (0.00)	mfcc6_1	0.17 (0.07)
DDmfcc6_1	0.12 (0.11)	mfcc10_2	0.12 (0.08)	lpcc3	0.1 (0.04)	lpcc9	0.17 (0.06)
plp8_1	0.11 (0.08)	DDmfcc4_1	0.12 (0.05)	lsf19	0.09 (0.09)	plp4_1	0.16 (0.04)
plpRast6_2	0.1 (0.05)	Dmfcc6_1	0.12 (0.07)	mfcc7_1	0.09 (0.04)	mfcc2_1	0.16 (0.11)
ZeroCross	0.1 (0.1)	lpcc3	0.11 (0.06)	mfcc6_1	0.08 (0.05)	DDmfcc11_1	0.15 (0.05)

From the plots in Fig. 1 it is clear that certain features are chosen more consistently by high performing classifiers than others. In each system investigated there is a strong peak at feature number 218. We can see from Table 5 that this corresponds to Inharmonicity. If a sound is perfectly internally 'harmonious' each of the upper partials will be integer multiples of the fundamental frequency. Inharmonicity is a measure of how much the spectral content of a sound differs from this ideal relationship. Although it has been generally used as a musical descriptor, its prominent and consistent selection in high performing classifiers in these experiments indicate that it may be a very strong indicator for voice verification also. Other individual spectral features are not strongly represented although the Zero Crossing Rate, Spectral Centroid and the Number of Spectral Peaks did appear in the top 20 features chosen by at least one system.

From Table 5 we can see that higher order PLPs were the next most selected feature. Within these only the first PC was chosen, indicating that the variance in the principle dimension for these features contains the most useful information. Surprisingly, the RASTA variations were not selected as frequently implying that the original implementation of the PLPs are more important for this problem. This may be because we used the high quality audio signal from the TIMIT database without adding noise. The RASTA method was developed to compensate for noisy channels, but as our signals are not noisy they are not found to be more beneficial than the standard PLP implementation.

The LPCCs were prominent among the highly selected features. LPCC3 was within the top 20 for each system and LPCCs 7 and 5 also featured in three of the four systems. Interestingly, the LPCs did not feature as strongly as their cepstral counterparts, indicating that in linear prediction for these problems the cepstral domain may be influential than the spectral domain. MFCCs have for a long time been one of the most widely used features in speech analysis. It may be surprising then to see that they did not appear as prominently as other features already discussed. In saying that, the first PC of a number of higher MFCCs did emerge as consistently chosen by successful classifiers. The derivatives of the MFCCs were among the least successful features.

6 Conclusions and Future Work

In this work we applied GP to evolve speaker verification programs on highly unbalanced training datasets. We found that using a number of independent runs, it is possible to evolve good-generalising programs, however good generalisation is not consistent in terms of median performance across all runs for the majority of systems. The MSE-based loss function that measures the discrepancy between program output and target value attained a median generalisation performance that is at least as good as the "majority classifier" for all speakers. This outperformed the loss function based on the weighted accuracy between minority and majority classes for most speakers. In addition, the MSE-based loss function performed better when used on the original unbalanced dataset than when used in combination with down-sampling in nearly all speakers. Finally, the use of non-equal weighted misclassification costs for the minority and majority classes did not significantly improve generalisation compared to an equal weighting.

In future work we plan to improve on the overfitting problem encountered during training. Restricting the size of the evolved solutions did not provide an effective remedy. We are experimenting with restricting the complexity of programs in the ST system through the use of a first-order Tikhonov-based regulariser that penalises functions that change rapidly. The Tikhonov function is minimised through multi-objective optimisation. Another possible solution against overfitting is the use of validation-based model selection for designating the output of the evolutionary process. Holding-out a validation set is however prohibitive using the limited training resources currently available. The use of analytical model selection methods that estimate the optimism of training error in terms of program complexity and training sample size is a possible line of attack. Having highlighted the limitations of standard GP fitness functions and data-sampling methods, we plan to experiment with additional fitness functions that were shown to be very effective in unbalanced datasets [4]. The use of the Area Under Curve as a fitness function is expected to drastically improve performance. We will compare these new experiments against current state of the art machine learning methods.

The feature selection reported is averaged across all successful speakers for each method. Our next experiment will evolve classifiers with GP using only the

subset of top 20 features selected in this study. This will determine not only what features to use, but the best way to combine them for robust ASV. Furthermore, we will consider the selection of features by individual speakers to investigate the dependency of feature extraction on individual voice characteristics.

Acknowledgments. This work was carried out as a collaboration of projects funded by Science Foundation Ireland under grant Grant Numbers 08/SRC/FM1389 and 13/IA/1850.

References

1. Agapitos, A., Brabazon, A., O'Neill, M.: Controlling overfitting in symbolic regression based on a bias/variance error decomposition. In: Coello, C.A.C., Cutello, V., Deb, K., Forrest, S., Nicosia, G., Pavone, M. (eds.) PPSN 2012, Part I. LNCS, vol. 7491, pp. 438–447. Springer, Heidelberg (2012)
2. Batista, G., Prati, R.C., Monard, M.C.: A study of the behavior of several methods for balancing machine learning training data. SIGKDD Explor. Newsl. **6**(1), 20–29 (2004)
3. Batista, G.E.A.P.A., Prati, R.C., Monard, M.C.: Balancing strategies and class overlapping. In: Famili, A.F., Kok, J.N., Peña, J.M., Siebes, A., Feelders, A. (eds.) IDA 2005. LNCS, vol. 3646, pp. 24–35. Springer, Heidelberg (2005)
4. Bhowan, U., Johnston, M., Zhang, M.: Developing new fitness functions in genetic programming for classification with unbalanced data. IEEE Trans. Syst. Man Cybern. B Cybern. **42**(2), 406–421 (2012)
5. Bhowan, U., Johnston, M., Zhang, M., Yao, X.: Evolving diverse ensembles using genetic programming for classification with unbalanced data. IEEE Trans. Evol. Comput. **17**(3), 368–386 (2013)
6. Burton, D.: Text-dependent speaker verification using vector quantization source coding. IEEE Trans. Acoust. Speech Signal Process. **35**(2), 133–143 (1987)
7. Campbell, W.M., Sturim, D.E., Reynolds, D.A.: Support vector machines using gmm supervectors for speaker verification. IEEE Signal Process. Lett. **13**(5), 308–311 (2006)
8. Curry, R., Lichodzijewski, P., Heywood, M.I.: Scaling genetic programming to large datasets using hierarchical dynamic subset selection. IEEE Trans. Syst. Man Cybern. B Cybern. **37**(4), 1065–1073 (2007)
9. Day, P., Nandi, A.K.: Robust text-independent speaker verification using genetic programming. IEEE Trans. Audio Speech Lang. Process. **15**(1), 285–295 (2007)
10. Dehak, N., Kenny, P.J., Dehak, R., Dumouchel, P., Ouellet, P.: Front-end factor analysis for speaker verification. IEEE Trans. Audio Speech Lang. Process. **19**(4), 788–798 (2011)
11. Doucette, J., Heywood, M.I.: GP classification under imbalanced data sets: active sub-sampling and auc approximation. In: O'Neill, M., Vanneschi, L., Gustafson, S., Esparcia Alcázar, A.I., De Falco, I., Della Cioppa, A., Tarantino, E. (eds.) EuroGP 2008. LNCS, vol. 4971, pp. 266–277. Springer, Heidelberg (2008)
12. Eggermont, J., Eiben, A.E., van Hemert, J.: Adapting the fitness function in gp for data mining. In: Langdon, W.B., Fogarty, T.C., Nordin, P., Poli, R. (eds.) EuroGP 1999. LNCS, vol. 1598, pp. 193–202. Springer, Heidelberg (1999)

13. Garofolo, J.S., Lamel, L.F., Fisher, W.M., Fiscus, J.G., Pallett, D.S.: Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1–1.1. NASA STI/Recon technical report n **93**, 27403 (1993)
14. Gathercole, C., Ross, P.: Dynamic training subset selection for supervised learning in genetic programming. PPSN III. LNCS, vol. 866, pp. 312–321. Springer, Jerusalem (1994)
15. Gonçalves, I., Silva, S., Melo, J.B., Carreiras, J.M.B.: Random sampling technique for overfitting control in genetic programming. In: Moraglio, A., Silva, S., Krawiec, K., Machado, P., Cotta, C. (eds.) EuroGP 2012. LNCS, vol. 7244, pp. 218–229. Springer, Heidelberg (2012)
16. Hermansky, H.: Perceptual linear predictive (plp) analysis of speech. J. Acoust. Soc. Am. **87**, 1738 (1990)
17. Hermansky, H., Morgan, N., Bayya, A., Kohn, P.: Rasta-plp speech analysis technique. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 1992, vol. 1, pp. 121–124. IEEE (1992)
18. Holmes, J.H.: Differential negative reinforcement improves classifier system learning rate in two-class problems with unequal base rates. In: 3rd Annual Conference on Genetic Programming, pp. 635–642. ICSC Academic Press (1998)
19. Huang, X., Acero, A., Hon, H.W., et al.: Spoken Language Processing, vol. 15. Prentice Hall PTR, New Jersey (2001)
20. Kenny, P., Boulianne, G., Ouellet, P., Dumouchel, P.: Factor analysis simplified. In: Proceedings of ICASSP, vol. 1, pp. 637–640. Citeseer (2005)
21. Kenny, P., Boulianne, G., Ouellet, P., Dumouchel, P.: Joint factor analysis versus eigenchannels in speaker recognition. IEEE Trans. Audio Speech Lang. Process. **15**(4), 1435–1447 (2007)
22. Kinnunen, T., Li, H.: An overview of text-independent speaker recognition: from features to supervectors. Speech Commun. **52**(1), 12–40 (2010)
23. Kubat, M., Matwin, S.: Addressing the curse of imbalanced training sets: one-sided selection. In: Fisher, D.H. (ed.) Proceedings of the Fourteenth International Conference on Machine Learning (ICML 1997), Nashville, Tennessee, USA, July 8–12, 1997, pp. 179–186. Morgan Kaufmann (1997)
24. Lartillot, O., Toiviainen, P.: A matlab toolbox for musical feature extraction from audio. In: International Conference on Digital Audio Effects, pp. 237–244 (2007)
25. Liares, L.R., Garca-Mateo, C., Alba-Castro, J.L.: On combining classifiers for speaker authentication. Pattern Recogn. **36**(2), 347–359 (2003)
26. Logan, B., et al.: Mel frequency cepstral coefficient for music modelling. In: ISMIR (2000)
27. Loughran, R., Walker, J., O'Neill, M., McDermott, J.: Genetic programming for musical sound analysis. In: Machado, P., Romero, J., Carballal, A. (eds.) EvoMUSART 2012. LNCS, vol. 7247, pp. 176–186. Springer, Heidelberg (2012)
28. Makhoul, J.: Linear prediction: a tutorial review. Proc. IEEE **63**(4), 561–580 (1975)
29. Márquez-Vera, C., Cano, A., Romero, C., Ventura, S.: Predicting student failure at school using genetic programming and different data mining approaches with high dimensional and imbalanced data. Appl. Intell. **38**(3), 315–330 (2013)
30. O'Shaughnessy, D.: Speech communication: human and machine. Digital Signal Processing. Addison-Wesley, Reading (1987)
31. Reynolds, D.A., Quatieri, T.F., Dunn, R.B.: Speaker verification using adapted gaussian mixture models. Digital sig. process **10**(1), 19–41 (2000)
32. Sivaram, G.S., Thomas, S., Hermansky, H.: Mixture of auto-associative neural networks for speaker verification. In: INTERSPEECH, pp. 2381–2384 (2011)

33. Song, D., Heywood, M.I., Zincir-Heywood, A.N.: Training genetic programming on half a million patterns: an example from anomaly detection. IEEE Trans. Evol. Comput. **9**(3), 225–239 (2005)
34. Winkler, S.M., Affenzeller, M., Wagner, S.: Advanced genetic programming based machine learning. J. Math. Model. Algorithms **6**(3), 455–480 (2007)