Evaluation of Objective Features for Classification of Clinical Depression in Speech by Genetic Programming

Juan Torres[1], Ashraf Saad[2], Elliot Moore[1]

[1] School of Electrical and Computer Engineering
Georgia Institute of Technology, Savannah, GA 31407, USA.
juan.torres@gatech.edu, emoore@gtsav.gatech.edu
[2] Computer Science Department, School of Computing
Armstrong Atlantic State University, Savannah, GA 31419, USA.
ashraf@cs.armstrong.edu

Abstract. This paper presents the results of applying a Genetic Programming (GP) based feature selection algorithm to find a small set of highly discriminating features for the detection of clinical depression from a patient's speech. While the performance of the GP-based classifiers was not as good as hoped for, several Bayesian classifiers were trained using the features found via GP and it was determined that these features do hold good discriminating power. The similarity of the feature sets found using GP for different observational groupings suggests that these features are likely to generalize well and thus provide good results with other clinical depression speech databases.

1 Introduction

In studies related to the classification of clinical depression from a patient's speech [1, 2], a database containing speech samples from healthy and depressed subjects was collected and a vast set of features was extracted from the database. In order to facilitate the design of an efficient and robust classifier, it is then desirable to solve the feature selection (FS) problem in order to find a (preferably small) subset of features that maximizes classification performance. Because the size of the feature search space is 2^N, where N is the total number of features, an exhaustive search becomes computationally prohibitive even for moderate values of N. Therefore, for most practical problems, a non-exhaustive FS algorithm must be employed.

Many feature selection algorithms have been proposed and survey of FS under the framework of classification can be found in [3]. FS algorithms can be divided into *filter* and *wrapper* approaches, which differ in the nature of the objective function to be maximized (or minimized). Filter approaches utilize a computationally inexpensive function designed to approximately measure the discriminating ability of the feature subset under consideration, whereas wrapper approaches use the actual performance of the classification algorithm. Thus, while wrapper approaches are

more computationally expensive, since a classifier must be trained for every feature subset that is evaluated, they can also be more accurate. Certain FS algorithms, such as Branch and Bound [3], are said to be both non-exhaustive and *complete*, meaning that they are guaranteed to arrive at the optimal solution without exploring the entire search space. However, a necessary condition to ensure completeness in these algorithms is that the objective function be monotonic with respect to the size of the feature subset. This condition, at least for finite data sets, precludes the use of the wrapper approach and therefore invalidates the completeness property with respect to classification performance.

A Genetic Programming (GP) approach to classifier design was proposed in [4] and later refined in [5] to perform *simultaneous* feature selection and classifier design by means of a single evolutionary search (GPFS), thus removing the computational expense of classifier training associated with the wrapper approach while allowing classifier performance to be used as a fitness measure for FS. Although this algorithm (like its aforementioned predecessors) is not guaranteed to find a globally optimal solution, the reduced computational burden with respect to wrapper approaches allows the search procedure to evaluate a much larger number of classifiers, which can increase the amount of exploration of the feature space. GPFS has been shown to produce excellent results relative to previous FS methods both in terms of classification performance and feature subset size. The results obtained in [5] using the GENE dataset are particularly motivating. On this dataset, the algorithm achieved an average classification accuracy of 92.55% using an average of 10.45 features out of 7129. In this paper, we present the results of implementing a GPFS-like algorithm using *lilgp* [6] to find a small set of highly discriminating features from a clinical depression speech database. Once this feature subset is found, we then address the question of whether a Bayesian classifier, which is optimal with respect to certain assumptions on the features' probability distributions, can provide better performance over the classifiers designed by the GP's evolutionary search.

The paper is organized as follows. A description of the clinical depression speech database and of the features contained therein is given in Section 2. Section 3 provides an overview of Genetic Programming and how it may be used for feature selection. Section 4 consists of a brief introduction to Bayesian classification and to several quantization and Probability Distribution Function (PDF) estimation methods for handling continuous data. Feature selection and classification performance results are given in section 5. Concluding thoughts and directions for future work are offered in section 6.

2 Speech Features

The features used in this paper were obtained from a database populated with the speech data of 18 patients (9 male, 9 female) with no history of mental disorders, and 15 patients (6 male, 9 female) who were undergoing treatment for a depressive disorder at the time of the study [1]. The speech corpus for each speaker consisted of a single recording session of the speaker reading a short story. The 65 sentences

contained in the corpus were stored individually. In addition, male and female speech was analyzed separately.

A set of raw features related to vocal tract resonances, the glottal waveform, and the Teager FM component were extracted from each voiced speech frame (approximately 25-30ms in duration). The extraction of vocal tract and glottal features is based on the source-filter model of speech production [7], which approximates a speech utterance as the convolution of a glottal (vocal fold) excitation signal with an all-pole linear filter representing the resonant frequency response of the vocal tract. Linear Predictive Analysis (LPA) can be applied to a frame of speech in order to approximate the vocal tract filter, from which formant (i.e. resonant) frequencies and bandwidths can be estimated. If done carefully, deconvolution by the LPA filter results in a reasonable approximation of the glottal excitation signal, which can be used to estimate regions of glottal opening and closure. From these regions, several glottal ratio and timing features can be obtained. Further details on the vocal tract and glottal features contained in the database and their extraction are given in [1]. Teager FM Features were extracted using the algorithm given in [8], with the exception that we did not limit ourselves to measuring only the variation in the frequency modulation (FM) component, but instead subjected the Teager FM signal from each speech frame to the statistical measures in Table 2, to obtain a set of 8 raw features. Finally, features related to prosodics (pitch, energy contour, and speaking rate) were extracted from each voiced section of speech within an utterance. The extraction algorithms for prosodic features are described in [2]. Raw features were grouped into the 10 categories listed in Table 1.

Table 1. Raw Feature Categories

Pitch (PCH)	Glottal Ratios (GLR)
Energy Median Statistics (EMS)	Glottal Spectrum (GLS)
Energy Deviation Statistics (EDS)	Formant Locations (FMT)
Speaking Rate (SPR)	Formant Bandwidths (FBW)
Glottal Timing (GLT)	Teager FM (TFM)

For each gender, two separate observation groupings were considered. The first grouping (G1) divided the corpus into 13 observations of 5 sentences each while the second grouping (G2) divided the corpus into 5 observations of 13 sentences each. A set of statistical measures (Table 2) was computed for each feature across each sentence. An additional set of statistics was computed only for pitch and energy features and is given in [2]. The resulting features are denoted as *direct feature statistics* (DFS). Each direct feature statistic was then subjected to the same set of statistical measures listed in Table 2, this time across all sentences in an observation. The procedure resulted in a large vector (approximately 2000 in size) of *observation feature statistics* (OFS). To produce an initial reduction in the dimensionality of the feature space, statistical significance tests (using Analysis of Variance – ANOVA) were conducted. Features that did not meet a significance level of ($p < 0.001$) were discarded. Table 3 shows the resulting number of OFS as well as the number of observations per gender and observational grouping (G1, G2).

Table 2. Statistical Measures

Statistical Measure	Equation
Average (AVG)	$\frac{1}{N}\sum_{i=1}^{N} x_i$
Median (MED)	50^{th} *percentile*
Standard Deviation (STD)	$\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i-\bar{x})^2}$
Minimum (MIN)	5^{th} *Percentile*
Maximum (MAX)	95^{th} *Percentile*
Range (RNG)	*MAX – MIN*
Dynamic Range (DRNG)	$Log_{10}(MAX) - log_{10}(MIN)$
Interquartile Range (IQR)	75^{th} *percentile* – 25^{th} *percentile*

Table 3. Observations and Observation Feature Statistics (OFS) per Experiment

Gender	Experiment	Observations	OFS
Male	G1	195	724
	G2	75	298
Female	G1	234	1246
	G2	90	857

3 Genetic Programming and Feature Selection

Genetic Programming is an evolutionary computing method in which an initial population of computer programs, each individual program representing a potential solution to an optimization problem, is evolved by means of biologically-inspired operations that will generally improve the fitness of individuals over generations [9]. Each individual in the population is represented by a set of trees whose nodes are built from a set of functions *F* and a set of terminals *T* that are suitable to tackle the problem at hand. Within the context of classification from real-valued data, a suitable choice for *F* consists of the arithmetic operators {+, -, *, /}, where a protected division operator (x/0=0) is used to ensure closeness. The terminal set *T* was chosen as the set of real number constants in the range of [0, 10] and the OFS set. An advantage of using these node sets is that the resulting classifiers consist of simple expressions that can be easily understood at first glance. A method for evolving *N*-class classifiers using multiple GP trees has been presented in [4]. However, for the two-class problem under consideration only one tree per individual is necessary since

classification can be performed according to the arithmetic sign of the output value of a tree.

Evolution in GP is performed by a set of operators that include reproduction, crossover, and mutation. These operators use individuals in a given generation to produce a new (and hopefully improved) generation of individuals. During crossover, two parent individuals are selected, and two offspring that are similar to the parents are produced. These offspring are usually created by swapping randomly-selected subtrees across parents. In mutation, a node of a tree is selected and replaced by a new randomly-generated subtree. The modified tree is then copied over to the new generation. Reproduction simply copies a parent to the new generation without modification.

Individuals are selected as inputs for reproduction based on several possible selection schemes, most of which utilize the fitness of an individual as a criterion for selection. The selection schemes considered here are fitness selection, in which individuals are chosen stochastically with a probability that is proportional to their fitness, and tournament selection, in which a pool of individuals is selected at random and the fittest member within this pool is then chosen.

The work reported in [5] has resulted in a new set of fitness functions, reproduction operators and selection criteria that extend the basic classification GP algorithm so that feature selection is simultaneously performed as the classifier population evolves. The algorithm implemented herein is a 2-class version of the one reported in [5], with the main difference being that we use single tree classifiers. A brief overview of the algorithm will be presented below.

Feature selection in the GPFS algorithm begins during the generation of the initial population, where a feature set is randomly generated for each individual. A feature set of size r is chosen with probability p_r linearly decreasing with respect to r, so that the initial population will contain many individuals with small feature sets and few individuals with large feature sets. Once the initial population is created, it is evaluated and the fitness of each individual is computed. In our context, one obvious choice for a fitness function is classification accuracy. GPFS introduces a variation of this fitness function that is biased toward individuals that use smaller feature sets, given as

$$f_s = f(1 + ae^{-r/n}) \ , \qquad \textbf{(1)}$$

where f is the original normalized fitness function, a is the bias strength, r is the number of features used by the individual, and n is the total number of features. The value of a decreases linearly with generations, so that initial generations highly favor smaller feature sets, but later generations focus on maximizing classification accuracy. As a result of using this fitness function, feature selection is performed mostly in the first few generations. In the current implementation, a can be as much as 0.2, giving a maximum bias of 20%.

Finally, we use the two crossover operations introduced in [5]. Homogeneous crossover restricts the selection of a pair of parents to those who share the same feature set. Heterogeneous crossover permits parents to have different feature sets, but has been biased toward selecting parents that use similar features. In heterogeneous crossover, the first parent is selected using tournament selection. During selection of the second parent, the fitness function of each candidate is

augmented by a small amount that is proportional to the similarity between its feature set and the feature set of the first parent. This bias is also limited to a maximum of 20%, but its strength remains constant over all generations. During each reproduction phase, heterogeneous and homogeneous crossovers are chosen at random, with the probability of homogeneous crossover P_{hg} given as:

$$P_{hg} = \frac{gen}{M} , \quad (2)$$

where *gen* is the index of the current generation and *M* is the total number of generations. As such, the probability of using homogeneous crossover increases linearly from 0 to 1 with generations. Heterogeneous crossover is performed with probability $1 - P_{hg}$. As a result, new combinations of features are explored during the first GP generations, while the last few generations focus almost entirely on improving those classifiers that already use a good feature set.

4 Naïve Bayesian Classification

In a probabilistic classifier, the class *C* with the highest probability of occurrence given the current set of observations is selected. This decision requires knowledge of the posterior probabilities $P(C_j \mid X_i)$ for $j = 1..N$, where *N* is the number of classes, and X_i is the feature vector of the i_{th} observation. Using Bayes' rule we can estimate the posterior probabilities, as given in equation 3:

$$p(C_j \mid X) = \frac{p(X \mid C_j)P(C_j)}{p(X)} , \quad (3)$$

where $p(X \mid C_j)$ is the class-conditional probability density function for class *j* (also called the *likelihood function*) and $P(C_j)$ is its *a priori* probability. The denominator term is the same for each class and can be safely ignored. The *a priori* probabilities for each class are usually determined by empirical information, such as the relative frequency of occurrence of each class in nature. In the present context, it is assumed that an incoming patient is equally likely to be depressed or not depressed. Therefore, the *a priori* term can be ignored as well. The task of constructing a classifier is then reduced to estimating the likelihood functions for each class from the training data.

To estimate the likelihood functions from continuous data, it is necessary to either quantize the data or assume a known form for the underlying PDF. The naïve Bayes' rule [10] assumes independence between all features in the class-conditional distributions. It has been shown to work fairly well in practice, even in some cases where the data violates the independence assumption [11, 12]. Under this assumption, the likelihood function for class *j* can be expressed as:

$$p(X \mid C_j) = \prod_i p(x_i \mid C_j) , \quad (4)$$

where x_i is the i_{th} component of the feature vector X. Thus, the PDF or discrete probabilities for each feature can be estimated separately. In the following

subsections we discuss five quantization and PDF estimation methods, two of which violate the independence assumption of equation 4.

4.1 Uniform Bins

In this quantization method, the features are scaled individually so that they lie in the range of [0, 1], a histogram with N uniformly spaced intervals (bins) is computed for each feature and each class using the training data. The class-conditional probability can then be estimated as:

$$P(x_i = a \mid C_j) = f_{i,j}(bin(a)) \ , \qquad (5)$$

where $f_{i,j}$ denotes the normalized histogram for the i_{th} feature and j_{th} class, and the function *bin* maps the feature value a into the appropriate histogram bin. The parameter N negotiates a tradeoff between quantization bias and variance [12]. The optimum value for N with respect to the product of classification sensitivity and specificity was found for each experiment by exhaustive search and is given in Table 7.

4.2 Optimal Threshold

This method is similar to uniform bins with N=2, with the difference that the cutoff threshold between the two bins is chosen separately for each feature [13]. The optimum threshold for a feature is chosen as the one that maximizes the product of sensitivity and specificity based on classification solely according to that feature. During classification, class-conditional probabilities are computed as in Equation (5), but using a feature-specific bin_j function.

4.3 Gaussian Assumption

Here we estimate the PDF of each feature and each class as a 1-D Gaussian density function whose mean and variance are taken as the sample mean and (unbiased) variance of the training data. The likelihood function for each class can then be evaluated by direct application of Equation (4).

4.4 Gaussian Mixtures

The Gaussian Mixture Model (GMM) is a popular density estimation method in pattern recognition [10, 14]. Each likelihood function $p(X \mid C_j)$ is modeled as a weighted sum of multivariate Gaussian densities. This approach has the advantage that given a large enough number of mixtures, an arbitrary (and possibly correlated) PDF can be accurately modeled. Training a GMM model consists of estimating the mean, covariance matrix, and weight for each density. The expectation-maximization

(EM) algorithm is used to estimate these parameters iteratively. The EM algorithm needs to be initialized with a fixed number of densities and an initial guess for the parameters of each density. This initialization is performed by k-means clustering, where the number of clusters equals the number of densities. The initial mean and covariance matrices are computed from the training points in each cluster. Due to the limited number of observations in our datasets, we use diagonal covariance matrices and limit the number of mixtures to 3 for the G1 experiments and 2 for the G2 experiments in order to reduce the number of parameters to be estimated.

4.5 Multivariate Gaussian

In this method, each (class-conditional) likelihood function is modeled as a single multivariate Gaussian PDF with a full covariance matrix, which is computed from the (unbiased) sample covariance between features in the training data. Like the GMM, this method does not follow the naïve Bayes assumption.

5 Results

The GPFS algorithm was run for each of the experiments shown in Table 3 using the parameters listed in Table 4. Ten iterations of leave-one-out cross-validation (LOOCV) [14] were performed for each combination of gender and observation grouping. Each iteration of cross-validation consisted of a number of GPFS runs equal to the number of observations in the dataset, with a different single observation left-out of the training set during each run. Classification accuracy (Table 5) was computed as the ratio of correctly classified left-out samples across all GPFS runs. In addition, we provide sensitivity and specificity rates, which correspond to the percentage of correctly classified depressed (positive) and non-depressed (negative) samples, respectively. The average feature set size values for each experiment are also given in Table 5.

Table 4. GPFS Parameters

Parameter	Value
Crossover probability	0.80
Reproduction probability	0.05
Mutation probability	0.15
Prob. of selecting int./ext. node during crossover	0.8 / 0.2
Prob. of selecting int./ext. node during mutation	0.7 / 0.3
Tournament size	10
Number of generations	30 for G1 / 20 for G2
Initial height of trees	2-6
Maximum allowed nodes of a tree	350
Maximum height of a tree	12
Population size	3000 for G1 / 2000 for G2

Table 5. Average Classification Accuracy and Feature Set Size

Experiment	Male		Female		Mean
	G1	G2	G1	G2	
Classification Accuracy	71.2	71.3	84.9	82.2	77.4
Sensitivity	80.9	74.7	85.4	82.7	80.9
Specificity	64.8	69.1	84.4	81.8	75.0
Feature Set Size	18.5	15.3	16.1	14.2	16.0

An approximate ranking of features was obtained by computing the frequency with which a feature is selected in the final solution tree of a GPFS run (Fig. 1). The motivation here is that since the GPFS algorithm is stochastic in nature, given a sufficiently large number of runs, the frequency of feature selection in the final solutions should provide a good indication of the discriminating performance of that feature. The 10 most frequently selected features for each experiment are listed in Table 6, with features that appear in more than one experiment in italics. The large similarity in the sets of the top 10 features for the G1 and G2 experiments within the same gender provide a good indication that the GPFS algorithm is consistently selecting certain specific features. It can also be seen in Table 6 that while glottal waveform features appear prominently in the male experiments, we have a large number of energy contour statistics taking the top spots for female subjects.

The average classification accuracy of the final GPFS classifiers was not very high. It should be noted that in the male experiments, the classifiers show a bias toward the class with the smaller number of training samples (depressed), which is counter-intuitive. Nevertheless, the relatively poor performance of these classifiers should not lead to the dismissal of GPFS as a useful feature selection method. Even if GP is unable to consistently find excellent classifiers, the fact that certain features are selected disproportionately more frequently from a uniform initial feature subset population is still an indication of their discriminatory power. To validate this assertion, we used the 16 most frequently selected features for each experiment to train Bayesian classifiers. Because the true form of the probability distribution of the features is unknown, we trained a separate classifier using each of the likelihood estimation methods discussed in Section 4.

The results for all likelihood estimation methods are shown in Table 7. Leave-one-out cross-validation was used for all methods. In addition, because the final result from training GMMs with the EM algorithm depends on the initial k-means clustering, which is in turn randomly initialized, the best result out of 10 training episodes is reported. A few interesting results are worth mentioning. For all its simplicity, the uniform quantization method outperforms the optimal threshold method even in the Male-G2 experiment where the number of bins is 2 for both methods. This suggests that setting optimal quantization levels on a per-feature basis may not be beneficial when the features are combined. The uniform quantization method also obtained the best overall accuracy for the female experiments. For all experiments, the naïve Gaussian and GMM results were fairly close. On average, we obtained an improvement in classification accuracy of 18.5% for males (GMM) and 7.1% for females (uniform quantization) relative to the GPFS classifiers.

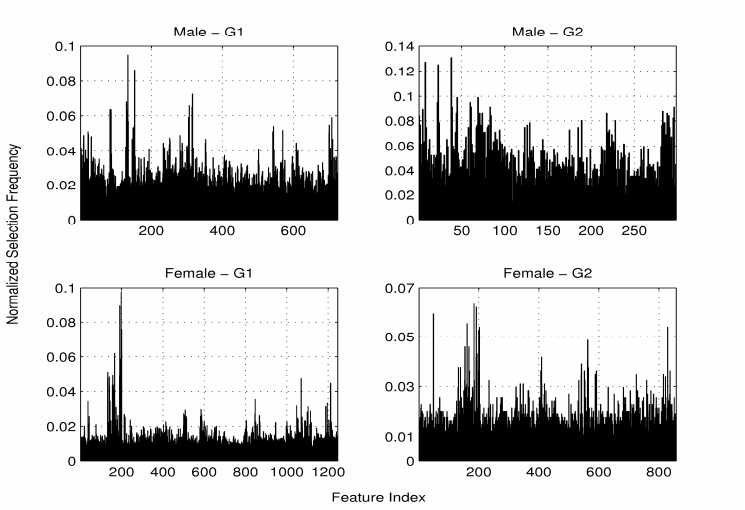

Fig. 1. Feature Selection Histograms

Table 6. 10 Best Features – Sorted by Rank

Male - G1	**Male - G2**
GLT: Max((CP)MIN)	*GLT: Max((CP)MIN)*
GLT: DRng((CP)IQR)	PCH: Med(A1)
GLS: Med((gSt1000)MAX)	*EDS: Avg(MED)*
GLT: Std((OP)IQR)	GLT: IQR((CP)IQR)
GLR: Rng((rCPO)IQR)	GLR: Min((rOPO)IQR)
GLS: Avg((gSt1000)MAX)	*EDS: Avg(AVG)*
EDS: Avg(AVG)	GLR: Med((rCPOP)MIN)
EDS: Avg(MED)	GLT: Std((CP)MIN)
EDS: Med(MED)	GLR: Max((rCPOP)MIN)
GLT: Med((CP)MIN)	*GLS: Avg((gSt1000)MAX)*
Female - G1	**Female - G2**
EMS: Med(MR)	EMS: IQR(AVG_1)
EMS: Med(STD_1)	*EMS: Med(STD_1)*
EMS: Max(MR)	PCH: IQR(IQR)
EMS: Med(RNG)	EMS: Med(STD)
EMS: Max(STD_1)	*EMS: Max(MR)*
EMS: Med(AVG)	TFM: Avg(MAX(IQR))
EMS: Max(MAX)	*EMS: Med(MR)*
EMS: Avg(STD_1)	FBW: Med((bwF3)IQR)
EMS: Avg(MED)	EMS: Med(MAX)
EMS: Avg(AVG)	*EMS: Med(RNG)*

Table 5. Bayesian Classification Performance

Exp	Method	Acc	Sen	Spec	Exp	Method	Acc	Sen	Spec
Male G1	Unif Bin (N = 8)	86.7	83.3	88.9	Female G1	Unif Bin (N = 9)	88.0	85.5	90.6
	Opt Thresh	82.6	82.1	82.9		Opt Thresh	78.6	65.8	91.5
	Gaussian	87.2	88.5	86.3		Gaussian	87.2	91.5	82.9
	GMM	88.7	87.2	89.7		GMM	87.6	88.0	87.7
	MVG	84.1	83.3	84.6		MVG	85.5	83.8	87.2
Male G2	Unif Bin (N = 2)	90.7	93.3	88.9	Female G2	Unif Bin (N = 5)	93.3	93.3	93.3
	Opt Thresh	73.3	50.0	88.9		Opt Thresh	86.7	75.6	97.8
	Gaussian	89.3	93.3	86.7		Gaussian	91.1	95.6	86.7
	GMM	90.7	90.0	91.1		GMM	88.0	83.3	91.1
	MVG	86.7	80.0	91.1		MVG	92.2	86.7	97.8

6 Conclusion and Future Work

By applying the GPFS algorithm, we were able to find a small set of individual speech features that are useful discriminators of clinical depression, as validated by the high performance of the Bayesian classifiers that were trained on these features. However, the true goal of FS is the selection of an optimal *combination* of features. In the context of the present work, this would require that we measure not simply how often single features are selected, but instead how often groups of features are selected together. An algorithm for ranking groups of features based on their joint selection frequency is currently under investigation.

Another area to investigate is the convergence rate of the feature selection performed by GPFS. The current implementation is designed to converge in the first few generations, but it might be desirable to instead allow a larger amount of feature set exploration throughout the evolution process. A recent technique [15] involving the use of SOMs to control the amount of exploration of the search space may be useful in this respect.

References

1. E. Moore, M. Clements, J. Peifer, and L. Weisser, Comparing objective feature statistics of speech for classifying clinical depression. In *Proceedings of the 26th Annual Conf. on Eng. in Medicine and Biology*, pages 17-20, San Francisco, CA, 2004.
2. E. Moore, M. Clements, J. Peifer, and L. Weisser, Analysis of prosodic variation in speech for clinical depression. In *Proceedings of the 25th Annual Conf. on Eng. in Medicine and Biology*, pages 2849-2852, Cancun, Mexico, 2003.
3. M. Dash, H. Liu, Feature selection for classification. *Intelligent Data Analysis*, 1(3):131-156, 1997.
4. D. Muni, N. Pal, and J. Das, A novel approach to design classifiers using genetic programming. *IEEE Transactions on Evolutionary Computation*, 8(2):183- 196, 2003

5. D. Muni, N. Pal, and J. Das, Genetic programming for simultaneous feature selection and classifier design. *IEEE Transactions on Systems, Man and Cybernetics*, Part B, 36(1):106-117, 2006.
6. D. Zongker, and W. Punch, *Lilgp 1.01 User's Manual*. Genetic Algorithms and Research Application Group, Michigan State University, East Lansing, MI, 1998. http://garage.cse.msu.edu/software/lil-gp/index.html.
7. T.F. Quatieri, *Discrete-Time Speech Signal Processing*, Prentice Hall, Upper Saddle River, NJ, 2001.
8. G. Zhou, J. Hansen, J. Kaiser, Nonlinear feature based classification of speech under stress. *IEEE Transactions on Speech and Audio Processing*, 9(3):201-216, 2001.
9. J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, 1992.
10. R. Duda, P. Hart, and D. Stork, *Pattern Classification*. Wiley, New York, 2001.
11. C. Elkan, Naive Bayesian learning. *Adapted from Technical Report No. CS97-557*, Dept. of Computer Science and Engineering, University of California, San Diego, CA, 1997.
12. Y. Yang and G. Webb, On why discretization works for naïve-Bayes classifiers. In *Proceedings of the 16th Australian Joint Conference on Artificial Intelligence (AI)*, pages 440-452, Perth, Australia, 2003.
13. M. Wiggins, A. Saad, B. Litt, and G. Vachtsevanos, Genetic Algorithm-Evolved Bayesian Network Classifier for Medical Applications. In *Proceedings of the Tenth World Soft Computing Conference* , 2005.
14. S. Theodoridis, and K. Koutroumbas, *Pattern Recognition*. Academic Press, San Diego, CA, 1999.
15. H.B. Amor and A. Rettinger, Intelligent exploration for genetic algorithms: using self-organizing maps in evolutionary computation. In *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation (GECCO)*, pages 1531-1538, Washinton D.C, 2005.