Music Genre Classification Using GA-Induced Minimal Feature-Set

Sushobhan Nayak and Ankit Bhutani
Department of Electrical Engineering
IIT Kanpur, UP, India
snayak, ankitbhu@iitk.ac.in

***Abstract*—We propose a genetic algorithm-based feature-selection method for music genre classification that not only increases the efficiency of standard classifiers, but also reduces the feature space to a bare-minimum. While previous works have been more focused on finding *near-optimal* features devoid of noise, we go for a modified fitness function capable of finding both the near-optimal and the *near-minimal* feature subset for classification. In addition to an enhanced performance, our model can also reduce the computational load for ill-formed sets and has the flexibility to incorporate trade-offs between efficiency and computational load. We finally demonstrate that the modified GA is capable of bringing about an 80% reduction in the feature space dimension at similar classification rates.**

***Index Terms*—Genetic Algorithms, Feature-set Reduction, SVM, kNN Classifier, Genre Classification**

I. Introduction

Music genre is frequently used as the query of choice while browsing through the music databases over the Internet[1]. While research into acquiring suitable features and classifiers for genre classification is abundant([2], [3], [4], [5]), works involving *selection* of relevant feature-sets particular to the task at hand are sparse at best. Some examples of such work would be [6], [7], where techniques of PCA, FFS and BFS have been used effectively.

In our work, we investigate the possibility of a relevant and minimal feature-set-selection with the help of Genetic Algorithms (GA). GAs have been used before by [8] and [2] to efficiently classify music genres. While [2] has shown pretty impressive results (a 90% classification rate) using GA, 109 musical features and a hierarchical classifier, [8] has been able to get almost 60% classification rate with GA, a set of classifiers like kNN, SVM and a feature set of 30 features. These works, though impressive, aim only for an optimal feature-set devoid of noisy features(which tend to misclassify samples). Furthermore, they use a primitive GA fitness function, viz. the hit-rate, for the best feature-set selection, which might converge to a local minima instead of a global one [9]. In this work, we not only find the optimal features from a set of given features for a given classification problem, but we also minimize the *number* of features required for the task. Using a modified fitness function, that also eliminates the local-minima problem of the standard hit-rate one, we bring about a more than 80% reduction in the original feature set, and a 50% reduction in the feature-set derived from the original feature-set by the hit-rate-only fitness function. To reiterate, we haven't proposed a new method for classification, or a new method for feature generation. We assume that the feature set and the classifier are defined by the task at hand. Given this scenario, and the fact that the standard GA, or other feature reduction techniques have already been employed to reduce the original feature-set, can we further reduce the number of features with minimal loss in performance? – this's the question we are going to handle in this paper.

The paper is organized as follows. In Section II, we give a brief description of the established GA and feature selection methods in literature, and propose our modification.Then in Section III, we describe the dataset, the feature-set and classifiers we are using for the experiment. Then in Section IV, we discuss the findings of our test and validate our claims. Then we conclude our discussion in Section V.

II. Proposed Model

Given a set of possible features and a classifier for a classification task, our objective is to derive an optimal feature-set devoid of noisy features with an aim of minimizing its cardinality.

Using GAs to derive near-optimal feature-subsets was brought to light by [10], and was further investigated in [11], [9]. In the present work, extending [10] and [9], we propose a modified GA to derive a near-optimal-cum-*minimal* feature-subset for music genre classification.

GAs emulate human evolutionary characteristics, and follow the survival-of-the-fittest paradigm. In an operationalized model of GA, we have a population of *chromosomes* (essentially a bit string), which are assigned *fitness values* (their relative worth) based on the task at hand. Fit individuals/chromosomes mate with each other, and that is simulated by *crossover*, in which they exchange parts of their chromosome. To account for sudden characteristic changes in some individuals of a population, *mutation* is simulated through bit flipping random parts of the chromosome. The fit individuals are selected from a population using some selection rule and the next generation consists of these fit individuals whereas unfit ones are discarded. With this approach, the GA claims that at the end of hundreds of generations, only the best chromosomes for the task at hand will remain, and that is the assumption based on which best features are selected. The crossover rate, mutation rate, selection rule, fitness function etc. are determined by the problem at hand.

978-0-7695-4599-8/11 $26.00 © 2011 IEEE
DOI 10.1109/NCVPRIPG.2011.61

For an n-dimensional feature space, we choose n-bit chromosomes. Following [10], a bit-value of '1' means that the feature-dimension is considered for classification, and a value of '0' means that the feature is eliminated from consideration. [1] The traditional fitness-function is the hit-rate (HR), i.e. the fraction of train-set samples that were correctly classified using the selected features. But such a fitness function has a tendency to converge to a local minima, and it also disregards the objective of creating the maximum separation between the classes under consideration. So, following [9], we use the following fitness function:

$$Fitness = HR + \gamma(CD/p)$$

where CD is the *class distance*, p is the total number of samples, and γ is a scaling constant. The class distance is defined as follows,

$$CD = \sum_{i=1}^{p}\sum_{j=1}^{p}(k \times SD_{ij})$$

where $k = -1/+1$ depending on whether i and j belong to the same class or not, respectively, and ,

$$SD_{ij} = \sqrt{\sum_{m=1}^{n} W_m(S_{im} - S_{jm})^2}$$

where, W_m is the weight$(0/1)$ of the m^{th} feature, and S_{im} is the value of the mth input feature in the ith sample. This ensures class separation between the training classes.

However, as we might notice, even if this fitness function is suitable for the task of finding *optimal* features for a classification task, it is inadequate for a *minimal* subset selection, because it incorporates no metric to reflect the property of a feature-set-minimization-task. To further reduce the number of features being used for classification, we, therefore, propose the following modification to the above fitness function:

$$Fitness = HR + \gamma(CD/p) + \beta(c/p)$$

where c is the number of zeros in the chromosome and β is a scaling factor. By including the number of zeros in a fitness function, we are favoring the individuals that use a minimum number of features, whereas their efficiency is taken care of by the HR and CD parts. The new proposed fitness function is thus expected to minimize the number of overall features used - at a similar efficiency. Simultaneously, it also allows us to trade between a higher efficiency or a lower computational cost (by using fewer features), through tweaking of γ and β. As we will presently see, the modification results in an almost 80% reduction in features from that of the bare classifier without GA, and a 50% reduction from the feature set derived using only HR and CD.

[1]Weighted features have been used in literature [12]; but they entail heavy computational load, and are not exactly helpful for the task at hand as we are trying to minimize the number of features used, not to increase efficiency by biasing the role of one feature over the other.

III. Experiment

A. Feature Set

We used audio features described in [13]. Each music file is represented in a 74-dimensional feature space. The features were extracted using J-Audio tool ([14]).

B. Dataset

We used part of the dataset described in [15]. We took the classes of *folk, jazz* and *rap* into consideration, which have more than 200 sample files each. The files were randomly divided into test and train sets for cross-validation.

C. Classifier

To demonstrate the working of our model, we used the simple SVM (linear kernel) and the k-nearest neighbour(5 neighbours) classifiers. The idea behind using them is that they have been used in abundance in literature, both for music classification ([13], [2], [3]) and GA-based studies([9], [10], [12]). This gives us a benchmark to compare our proposition with the existing performance-statistics.

A number of experiments were done with the following being the variables:

- Number of samples in the train set (50/100/150 per class for the 3-class classification)
- Classifier (kNN/SVM)
- Fitness function (Basic HR/ HR and CD/ HR, CD, and Zero-count(ZC)/HR and ZC)
- Scaling factor β (0.5/1)

For the GA implementation, following [9], we set the crossover and mutation probabilities to 0.6 and 0.05 respectively. Parameter γ was set to 0.05. The selection of the best specimens was done through Roulette Wheel selector, and the algorithm was run for 100 generations with a population of 80 on each run. GA was implemented in Python programming language, using the *PyEvolve* toolkit[16]. The classifiers were also implemented in Python, using the *scikits.learn* toolkit[17].

IV. Results and Discussion

To recapitulate, we started with a 74-D feature space, and the objective was to investigate if the proposed fitness function can reduce the feature space to a minimum, without incurring heavy penalty on the performance of a given classifier on a given training and test set. We see from Table I [2], that without employing GA, the classification rate hovers around 50%, which is close to the results found in [13] on a different dataset using the same features. We also notice that the basic GA with the hit-rate only fitness function, increases the classification rate by more than 20% for 50/100 sample train sets. But, the features it uses to arrive at such a rate are > 40 in number. We see that it is still very efficient (using only 50% of the

[2]HR = hit-rate only. CD = Class Distance. ZC = Fitness function with zero-count. feat.= no of features used, i.e. the number of ones in the best individual/chromosome. [1/0.5] = The value of the parameter when $\beta = 1/0.5$. So, 17/29 means 17 features were used when $\beta = 1$, whereas 29 were used for $\beta = 0.5$ for the given classification rate. HR and HR+CD used same number of features, i.e CD was ineffective for this ill-formed set.

TABLE I
TABLE SHOWING THE EFFECT OF VARIOUS PARAMETERS ON CLASSIFICATION RATE AND NUMBER OF FEATURES REQUIRED, FOR THE **SVM** CLASSIFIER.

No. of Samples	rate(no GA)	rate(HR)	rate(HR+CD)	rate(HR+CD+ZC)[1/0.5]	feat(HR+CD)	feat.(HR+CD+ZC)[1/0.5]
50	47.74	64.86	64.86	69.36/75.07	47	17/29
100	46.64	70.27	70.27	70.27/68.46	42	27/38
150	54.05	54.05	54.05	57.65/54.05	42	24/36

TABLE II
TABLE SHOWING THE EFFECT OF VARIOUS PARAMETERS ON CLASSIFICATION RATE AND NUMBER OF FEATURES REQUIRED, FOR THE **KNN** CLASSIFIER.

No. of Samples	rate(no GA)	rate(HR)	rate(HR+CD)	rate(HR+CD+ZC)[1/0.5]	feat(HR+CD)	feat.(HR+CD+ZC)[1/0.5]
50	54.04	54.04	54.04	50.45/51.35	44	12/17
100	50.45	50.45	50.45	42.34/45.94	42	13/22
150	50.45	46.84	46.84	46.84/47.74	41	18/16

TABLE III
TABLE SHOWING THE BEST INDIVIDUAL IN **KNN** RUNS. THE 1'S ARE THE FEATURES THAT ARE USED FOR CLASSIFICATION.

No. of Samples	Best Individual
50	00000000000101000000110000010000000001001101001000000000010000001000000000
100	01000000000100100100001000001000000000000010001001000100011000000000100000

original feature dimensions), and has been able to eliminate any noisy or unnecessary features. But can we reduce the dimensions further? We see that with our modified fitness function, this indeed is possible. With the modified function, same or better success rate can be arrived at with just close to 20 features, which is a more than 50% reduction (from 47 to 17) at the same success rate, i.e. we have been able to reduce the number features used for classification without compromising on efficiency. In fact, the efficiency of $> 60\%$ classification in music genre classification without post/pre-processing is very rarely surpassed. While [8] barely reaches the 60% mark in classification, [13] asserts a 47% success rate by raw SVM (which is later increased to 75% after some post processing). [2] however has a 90% classification rate, but that too after repeated use of GA at different stages of a hierarchical classification. While we set out with a 74-D feature space, we have arrived at a better classification rate with just 20-something features

While the modified algorithm works very well with the SVM classifier, we don't see the same trend with kNN. When kNN is used as the classifier of choice, our algorithm fails to improve the classification rate; in fact, it causes it to drop by 4-5%. However, one might notice that the basic GA, with the hit-rate only fitness-function, also fails to improve the rate. On the other hand, while basic GA requires 41-44 features to arrive at the said success rate, our algorithm uses only 12-18 features to reach at its optimum classification rate – an almost 75% reduction in feature-space dimension. This has a rather interesting implication. As is suggested by the performance of basic GA, some classifier, feature and training set combinations might be adamant to any improvement using non-linear feature reduction techniques like GA (*ill-formed sets*). In that case, for that set of classifiers and training samples for which efficiency improvement is a Herculean task, the best one can do is to reduce the computational effort by reducing the feature space. And our proposed model does a very good job at that.

Table III shows some of the best individuals. In case of the kNN classifier, the features most often used are Compactness overall standard deviation and average, Strongest Frequency Via Spectral Centroid Overall Standard Deviation and Strongest Frequency Via Zero Crossings Overall average. The average and standard deviation of only the first MFCC coefficient is generally used while others are very rarely used. SVM, on the other hand, rarely uses the standard deviations and averages of Strongest Frequency via Zero Crossings Overall, via Spectral Centroid Overal, via FFT Maximum Overall and Compactness Overall average, thereby showing a marked difference of choice between the classifiers. Only 4 of the MFCC coefficients are primarily used by SVM, and the fact that [18] supports this observation (only 5 of the MFCCs are important) adds strength to our model. Furthermore, usually, both the average and standard deviations of any feature are either considered or not considered for classification.

The effect of varying β is also visible. By increasing β we are biasing the algorithm to look for smaller feature-sets at the expense of efficiency, while decreasing it essentially means that we are more biased towards efficiency. From Table II we notice that, changing β from 1 to 0.5 results in a consistent increase in both performance and the number of features used for classification, as was expected. However, in case of SVM, we notice that similar consistency is not shown. While in the first case performance increases as expected, the latter two runs show a 2-3% drop in success rate. The anomaly hardly undermines our contribution though, which was to propose a new model that might reduce the feature space without serious performance issues, and even without a detailed study of parameter-variation-effects, we have been

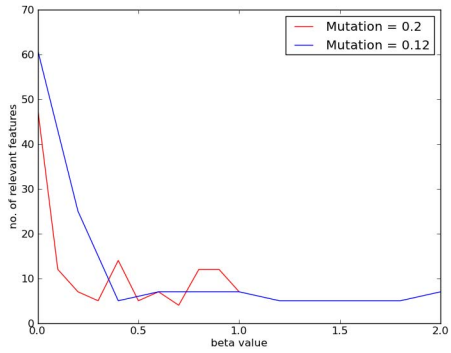

Fig. 1. No. of relevant features against variation of β for kNN classifier

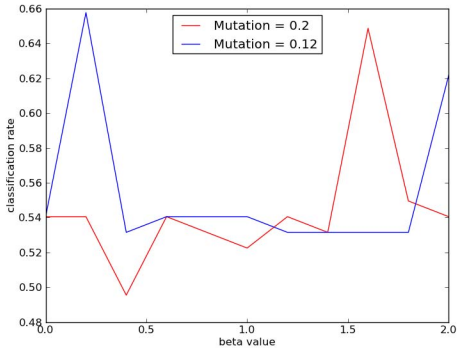

Fig. 2. Classification rate against variation of β for kNN classifier

able to demonstrate as much.

Nonetheless, to investigate the possibility that there might be an optimal β for each classification task, further experiments were done with different combinations of mutation, crossover rate and β, on both the classifiers. Fig. 1 and 2 show the plots for two such tests (2-point crossover prob. of 0.8 and varying mutation rates) involving the kNN classifier. Fig. 1 points out that after a certain threshold, the reduction in β has no visible effect on the reduction of feature space, which is natural because the feature space has already been reduced by $> 85\%$ (less than 10 features used, with the minimum being a classification rate of $> 50\%$ with only 5 features) and anymore reduction can only lead to heavy penalty in the classification rate, which is prevented by the CD and HR part of the fitness function. Consequently, the classification rate remains almost the same(Fig. 2), because the same number of features are being used, with occasional peaks that can be attributed to mutational anomalous convergence.

V. Conclusion

In this study, we showed that using basic GA, we can reduce the feature space and enhance efficiency of music genre classifiers. We then went on to propose a modified fitness function that can further reduce the feature-space without incurring performance-drop. Using the new GA, we have shown that the same classification rate can be achieved using almost only 20% of the feature space we set out with. Besides increasing classification rate, our model is also capable of reducing computing time for ill-formed sets. The model also allows for a trade-off between efficiency and computational cost by varying the inherent parameters. A rigorous treatment of the same, however, has not been undertaken in the present work. While preliminary data seems supports our prediction, a mathematical insight into the effect of β on the convergence of the algorithm and other anomalies is expected to be explored in future investigations.

References

[1] J Lee and J Downie. Survey of music information needs, uses, and seeking behaviours: preliminary findings. In *Proc. of the 5th Intern. Conf. on Music Information Retrieval*, pages 441 – 446, 2004.

[2] Cory McCay and Ichiro Fujinaga. Automatic genre classification using large high-level musical feature sets. In *Proc. of the International Conference on Music Information Retrieval*, pages 525–530, 2004.

[3] Cory McCay and Ichiro Fujinaga. Automatic music classification and the importance of instrument identification. In *Proc. of the Conference on Interdisciplinary Musicology*, 2005.

[4] T. Li and M. Ogihara. Music genre classification with taxonomy. In *Proc. of IEEE Intern. Conf. on Acoustics, Speech and Signal Processing*, pages 197 – 200, 2005.

[5] Carlos N. Silla Jr., Celso A. A. Kaestner, and Alessandro L. Koerich. Automatic genre classification of latin music using ensemble of classifiers. In *Proc. of the 33rd Intigrated Software and Hardware Seminar*, pages 47 – 53, 2006.

[6] R. Fiebrink and I. Fujinaga. Feature selection pitfalls and music classification. In *Proc. of the 7th Intern. Conf. on Music Information Retrieval*, pages 340–341, 2006.

[7] Y. Yaslan and Z. Cataltepe. Audio music genre classifica- tion using different classifiers and feature selection methods. In *Proc. of the Intern. Conf. on Pattern Recognition*, pages 573 – 576, 2006.

[8] Carlos N. Silla Jr., Alessandro L. Koerich, and Celso A. A. Kaestner. Feature selection in automatic music genre classification. In *Proc. of the Tenth IEEE International Symposium on Multimedia*, pages 39–44, 2008.

[9] S.H. Kim and S.W. Shin. Identifying the impact of decision variables for nonlinear classification. *Expert Systems With Applications*, 18:201 – 214, 2000.

[10] W. Siedlecki and J. Sklansky. A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, 10:335 – 347, 1989.

[11] W.F. Punch and E.D. Goodman et al. Further research on feature selection and classification using genetic algorithms. In *Proc. of the Int. Conf. on Genetic Algorithms*, pages 557–564, 1993.

[12] JD Kelly Jr. and L. Davis. A hybrid genetic algorithm for classification. In *Proc. of the International Joint Conference on Artificial Intelligence*, pages 645–650, 1991.

[13] Yi Liu, JiePing Xu, Lei Wei, and Yun Tian. The study of the classification of chinese folk songs by regional style. In *Proc. International Conference on Semantic Computing*, pages 657–662, 2007.

[14] Daniel Mcennis, Cory Mckay, and Ichiro Fujinaga. Jaudio: A feature extraction library. In *International Conference on Music Information Retrieval*, 2005.

[15] H Homburg, I Mierswa, B Moller, K Morik, and M Wurst. A benchmark dataset for audio classification and clustering. In JD Reiss and GA Wiggins, editors, *Proc. of the Int. Symposium on Music Information Retrieval*, pages 528–531, 2005.

[16] CS Perone. Pyevolve. http://pyevolve.sourceforge.net/.

[17] scikits.learn. http://scikit-learn.sourceforge.net/stable/.

[18] M. Noris, D. Shyamala, and W. Rahmita. Factors affecting automatic genre classification: an investigation incorporating non-western musical forms. *Proc. ISMIR*, 2005.