Audio Feature Selection and Classification

Bernie Sander
Computer Science
Northeastern University
Portland, ME

Dylan Wu
Computer Science
Northeastern University
Portland, ME

Jackie Himel
Computer Science
Northeastern University
Portland, ME

***Abstract*—Classifying musical audio files is an important task, given how many unclassified files are uploaded online every day. This project focuses on two main classification tasks: classifying audio as music or non-music and classifying music as one of 10 musical genres. We use K-Nearest Neighbor and Neural Network models to perform these tasks, using a feature vector composed of the means and variances of MFCCs, Spectral Centroid, and Zero Crossing Rate, and tempo features from WAV files using the Librosa Python library. We then optimize the input feature vector by running a genetic algorithm on a pool of many different feature vectors, and evaluating the fitness of each of the features on both of our ML models. Although we record high accuracy for both classification tasks using the feature vectors automatically selected by our genetic algorithm, further testing and analysis is required for statistical significance. This study serves to illustrate the process by which a genetic algorithm selects the optimal feature vector for machine learning algorithms.**

I. Introduction

Our problem has two main parts: classifying audio files using ML models and then selecting better subsets of features for those models using a genetic algorithm [2]. To examine differences in feature subsets chosen by the genetic algorithm, we create two audio-based classification tasks: classifying a WAV file as music or not and classifying a music WAV file into one of 10 genres (which are comprised of Rock, Pop, Classical, Metal, Reggae, Blues, Jazz, Country, Disco, Hip-hop). As is the case in any application of machine learning algorithms, sizeable and varied data sets are required. For the musical genre classification task, we used the GTZAN data set, which has been used in many previous research papers involving this classification task [6]. For our binary music/non-music classifier, we used a combination of three data sets containing sounds in the categories of speech, environment (like birds chirping) [3] and urban sounds (like a car) [4]. Finding an adequate collection of non-music sounds proved to be a more involved task, since the amount of diversity in this category is exceedingly vast. The details of these data sets will be discussed in more depth in the methodology section of this paper.

Our work consists of three main stages. The first involves downloading and cleaning the data sets and parsing the files containing the audio signals into the desired features. A component of this task was deciding which features to choose and why, to increase the accuracy of both classification tasks. Details of these features are discussed in the literature review of this paper.

The second part of this project involved setting up the machine learning models, I.e., KNN and a Neural Network, to accept as input the feature vectors, train on the feature vectors of segments of audio signals parsed from the files, and then test those trained models.

Once this task was complete, we designed a simple genetic algorithm that begins with a population of feature vectors, where each feature vector is evaluated by getting the accuracy of the models after training and testing them on that feature vector. Secondly, we performed a similar algorithm on our neural net. However, rather than changing the feature vectors with crossovers and mutation, we performed crossover and mutation operations on the number of layers and neurons in the neural net population. Many decisions had to be made with regard to the parameters of our genetic algorithm, e.g. how the populations would evolve from generation to generation, how mutation and crossover operations would work, how many generations we would have, and what the starting populations would be.

II. Literature Overview

There is a plenitude of research on the topic of feature selection for audio classification. This work varies widely in terms of methods used and types of feature vectors used for classification. For genre classification, the one constant in virtually all the research papers is the data set – GTZAN, a collection of data files for 10 different genres, 1000 samples (3 second segments) for each genre [5]. Based on the previous successes of this data set, as well as the factor of accessibility, we decided to use this data set for our genre classifier.

We chose our features based both on previous research as well as the qualities that they are said to represent in the audio. The most widely used feature seemed to be Mel Frequency Cepstral coefficients (MFCCs). As cited by Jingwen Zhang, this feature encapsulates the qualities that are perceived by the human ear when it listens to sound [12]. This feature is taken by first taking the Discrete Fourier Transform (DFT) of the signal, which converts the signal from the time domain to the frequency domain. On a high level, this is then converted into a Mel frequency spectrogram, which can be visualized as a plot of the frequency with respect to time, where each frequency "bin" is colored according to its loudness (measured in dB). The Mel part of the name means that the logarithm is taken of both the frequency and the loudness to account for the human logarithmic perception of sound [10].

Another feature we decided to use is the Spectral Centroid. This feature has been used to effectively predict the brightness and timbre of an audio signal [11]. Supplementing this feature, we used the Zero Crossing Rate (ZCR) to measure the amount of noise in the audio. ZCR has been used to classify the percussiveness of different sounds [8]. Since the genre of a song would seem to depend on the amount of percussion the song has, we figured that this would be a relevant feature to use, particularly for our genre classifier, although we use it for both classifications.

G. Tzanetakis et al classified 6 different genres in addition to binary classification of music and speech, using a 9-dimensional feature vector with the means and variances of spectral centroid, spectral rolloff, spectral flux, zero crossings, and low energy. This study, which used a Gaussian classifier for both classification tasks, yielded an accuracy for genres of 62% (as opposed to a random classification of 16%) and 86% for binary music and speech classification (as opposed to a random classification of 50%) [6].

A paper that uses the GTZAN data set to train a KNN model to classify 10 genres, reports an accuracy of over 80% for both feature vectors of size 5 and size 10 [7]. Note that their feature vector also uses MFCC mean and variance values.

Using the 4 genres from the GTZAN data set, and using MFCCs exclusively in their feature vector, Michael Haggblade et al report an overall accuracy of about 80% using KNN and 96% using a neural net [9].

We referred to these studies largely for inspiration on which features we would extract as a starting point, and then see what features the genetic algorithm decided to keep and discard in its optimal feature vector.

III. Methodology

We extracted data from four separate data sets. For genre classification, we used GTZAN data set, which contains 3000 seconds of audio for each of 10 genres. For binary audio classification, we used GTZAN paired with 3 non-music data sets: one containing speech, one contain environmental sounds, and another containing urban sounds. After segmenting each of the files into 3 second long clips, we had a total of 10k music clips and 10k other audio clips. This means that for training our models, we had 10k music clips for genre classification, and twice that for binary audio classification. All of these files were encoded as 16 bits per sample. Most of the files were mono, but for those that were in stereo, we converted them to mono using Librosa, to ensure consistency. After extracting the segments, we then parsed these samples into csv files containing the features. For features, we chose to have the means and variances of 20 MFCCs, Spectral Centroid, Spectral Rolloff, and tempo. We designed our code to make it easy to add more features, but for this project we chose to use these features only for the sake of simplicity.

We chose to use K-Nearest Neighbors (KNN) and Neural Net models for the reasons that the former is very simple and the latter very complex, and because they are both commonly used in audio classification. We use the Python library Scikit-learn for both models. However, for KNN, we also define a wrapper class to define our own fit and predict wrapper methods. Based on the literature as well as intuition, we expected Neural nets to perform better than KNN for both classification tasks.

Regarding our genetic algorithm, for each generation assuming a population of 100, the population is set as follows: we keep the 50 fittest parents and the 50 fittest children (I.e. the fittest half of the old population and the fittest half of the new population). Originally, we set the new population to be all the children, and none of the parents. We decided to keep the fittest parents in order to lean more towards local optima (at the cost of less diversity) and to speed up our algorithm as well, so that it would only have to iterate through half of the parents each generation.

In each generation, the parents reproduce children that are composites of new features and new parameters. For example, for both KNN models and Neural Nets, a genetic algorithm is run to optimize the feature vector, as well as the k value for KNN and the number of layers and neurons for Neural Nets. Therefore, the genetic algorithm returns the fittest individual, which contains a feature vector, a k value or number of layers/neurons, and an accuracy (or fitness).

Many decisions regarding mutation and crossover were made in designing our genetic algorithm. In the case of feature selection, irrespective of the inputted model, we were faced with several invariants with respect to how a feature vector is transformed between generations. Firstly, a feature vector must be a unique collection of features; there must not be any duplicate features. Secondly, the crossover operation must be fair on average; each child must inherit more or less equally form the the parents. Thirdly, the order of the features in the feature vector does not matter. This means, that in mutating a feature vector, we cannot swap two features as we would in the case of an application such as the traveling salesman problem. Instead, we select a random feature from our pool of features that does not already exist in the feature vector. It should be recognized that, while our algorithm follows the normal pattern of genetic algorithms, the reproduction step is unique in the sense that uniqueness of each gene must be maintained, and the uniqueness of the feature vector does not depend on the order of the features within in it.

The genetic algorithm differs slightly depending on the model fed into it. In the case of KNN, each individual in the initial population has a feature vector mapped to an accuracy and a k-value. Throughout the course of the algorithm, similar to feature selection as noted above, the k-value is crossed-over between the parents (by taking the average of the two k-values) and mutated, by increasing or decreasing the k-value by 1, determined by a set of probabilities.

The genetic algorithm for the neural net mirror that of the KNN except that instead of performing crossover and mutations on the k-value, it operates on the number of layers and neurons of both parents.

IV. Results

We tested our process of feature selection by running our genetic algorithm on the KNN model with the parameters of 100 individuals over a population of 50, for both classification tasks. For the neural net, we ran the algorithm on 50 individuals over 10 generations, since the neural net took more time to evaluate (i.e. train and test in the evaluation function of our algorithm).

For an initial population of 100 individuals over 50 generations using the KNN algorithm, our genetic algorithm outputted a feature vector of 47 features, with an accuracy of approximately 98% for binary classification and 89% for genre classification (10 genres). Interestingly, in both cases the genetic algorithm outputted a feature vector that has the maximum number of features. In other words, it used all the available features to come to its near "optimal" result. Both accuracy values are higher than the accuracy values we get without using the genetic algorithm to find the optimal set of features. Furthermore, while the existing literature uses other models and techniques, and sometimes limits the feature vector size, our models perform better than the accuracy values recorded in the aforementioned papers.

For our neural net, our genetic algorithm gave us an accuracy of 98% for binary classification, yielding a model with 2 layers and 31 neurons with the maximum feature vector size. For genre classification, the accuracy was approximately 81%, giving a neural net with 2 layers and 42 neurons.

As expected, we noticed an improvement when increasing the number of generations and individuals. For example, in the case of the neural net for binary classification, the accuracy increased from 96% (with 2 generations and a population of 2), to 98% accuracy (with a population of 50 over 10 generations).

One interesting pattern in all of our results is that, while each feature vector in the initial population starts out with a size between 8 and 36 features, it gathers more features over time, eventually outputting a feature vector that comprises all of the possible features.

V. Discussion

Initially, it appeared that our method of feature selection was effective because our accuracy values are quite high. However, we should note that much of our data samples are highly correlated with another, which could explain this very high accuracy. This is due to the fact that we split up the original musical files into 3 second segments in order to have more training data.

Unsurprisingly, the accuracy of both models on genre classification is much lower than that of binary classification. This, of course, makes intuitive sense, since there are fewer classes in binary classification. We should also caveat that, while our binary classification data is more diverse than just speech, with the inclusion of environmental and urban sound data sets, it is still unable to encompass general non-music sounds.

One possibly fruitful extension we could add to our genetic algorithm would be to tie in the accuracy value of an individual with the probability that it is mutated. For example, if we have a really high-accuracy individual, we would desire that there is a lower probability that this individual is mutated than one of a lower fitness. This change would reconfigure our algorithm to operate more in the style of simulated annealing, rather then its current mode of operation, which is more like stochastic hill-climbing.

Another consideration in amending our algorithm is to create more diversity in our initial feature vector population. There are many features that have been cited as being effective, that could provide more interesting results [1, 10, 12]. Unfortunately, we ran out of time to add this to our algorithm, but it would be worthwhile to conduct more analysis into how and why feature vectors are constructed, depending on the model used and the classification task at hand, and then to feed back those results into a more robust initial feature vector population.

VI. Conclusion

The confluence of three rich areas of research: machine learning algorithms (including deep learning), digital audio processing and feature extraction, and optimization of feature vectors and model parameters, provides a panoply of opportunities for significantly improving the accuracy of many music and general audio classification tasks, by reducing the amount of manual feature vector experimentation and instead placing feature selection in the hands of genetic algorithms. This project serves as a foray into this fascinating intersection of topics. Given further analysis and diversification of the data sets, more thoughtful and diverse selection of initial feature vectors, and further experimentation of genetic algorithm design, all through a more rigorous design and testing process, we hope that this initial research will someday yield significant results.

References

[1] Abdullah I. Al-Shoshan, Atiyah Alatiyah, Khalid Al-Mashouq, A Three-Level Speech, Music, and Mixture Classifier, Journal of King Saud University - Engineering Sciences, Volume 16, Issue 2, 2004, Pages 319-331, ISSN 1018-3639, https://doi.org/10.1016/S1018-3639(18)30794-3.

[2] J. Menezes, G. Cabral, B. Gomes. "Genetic Algorithms for Feature Generation in the Context of Audio Classification". World Academy of Science, Engineering and Technology, Open Science Index 110, International Journal of Electrical and Information Engineering (2016), 10(2), 427 - 430.

[3] https://www.kaggle.com/datasets/mmoreaux/environmental-sound-classification-50.

[4] J. Salamon, C. Jacoby and J. P. Bello, "A Dataset and Taxonomy for Urban Sound Research", 22nd ACM International Conference on Multimedia, Orlando USA, Nov. 2014.

[5] Marsyas. "Data Sets" http://marsysas.info/download/data_sets.

[6] G. Tzanetakis et al. "Automatic Musical Genre Classification of Audio Signals". https://ismir2001.ismir.net/pdf/tzanetakis.pdf

[7] D. Jang and S. Jang, "Very short feature vector for music genre classiciation based on distance metric lerning," 2014 International Conference on Audio, Language and Image Processing, 2014, pp. 726-729, doi: 10.1109/ICALIP.2014.7009890. https://ieeexplore.ieee.org/document/7009890

[8] F. Gouyon et al. "On the Use of Zero-Crossing Rate for an Application of Classification of Percussive Sounds". Proceedings of the COST G-6 COnference on Digital Audio Effects (DAFX-00), Verona, Italy, December 7-9, 2000.

[9] M. Haggblade et al. ”Music Genre Classification” http://cs229.stanford.edu/proj2011/HaggbladeHongKao-MusicGenreClassification.pdf.
[10] D. Raya et al. ”Music Genre Classification”. https://www.saibhaskardevatha.co.in/projects/dsp/index.html#4
[11] E. Schubert et al. ”Spectral centroid and timbre in complex, multiple instrumental textures”. Proceedings of the 8th International Conference on Music Perception & Cognition, Evanston, IL, 2004.
[12] Jingwen Zhang, ”Music Feature Extraction and Classification Algorithm Based on Deep Learning”, Scientific Programming, vol. 2021, Article ID 1651560, 9 pages, 2021. https://doi.org/10.1155/2021/1651560.