Music vs. Non Music Classifier for ISMIR 2020

Matthew Arnold
School of Music
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
Email: marnold@gatech.edu

***Abstract*—Our aim was to find a solution for segmenting audio files from Florida Band All-State auditions by identifying the times in which the musical exercises took place in the audio recordings. We calculate seven features from the audio: RMS, ZCR, Spectral Centroid, Spectral Rolloff, Spectral Crest Factor, Spectral Flux, and Spectral MFCC's. We then use these features to train a Support Vector Machine fit for classifying future inputs.**

I. Introduction

This problem is a single step in a larger project that utilizes neural networks in predicting the judges scores for All-State auditions given only the audio file of the audition and the digital sheet music for the exercises. The only sections of the audition that need to be evaluated by the network are the exercises themselves, so prior to our approach to solving this problem, the members of the project group were required to annotate the sections of the audio recordings manually. By training an SVM capable of classifying these sections, the parameters of annotation can be standardized and the rate at which these audio files can be annotated is greatly increased.

II. Method

Two primary steps are involved in our approach to this problem: (1) Extracting feature data and the ground truth vector, and (2) Training the SVM classifier for the testing of new data.

Finding meaningful features that were well fit for this problem was a very important step. The features we chose can be categorized into either Time Domain features, or Spectral features. The two Time Domain features are RMS and ZCR, and the inputs for these features are the audio signal, block size, hop size, and sample rate of the audio.

The five Spectral Features are Centroid, Rolloff, Crest Factor, Flux, and MFCC's. These features have common inputs as well, in that all of them use the Spectrogram of the audio signal and the sample rate of the recording.

A Spectrogram is a chronological mapping of individual Spectra which are calculated for each block of audio. The Spectra are obtained through Fourier Analysis to determine the magnitude of each frequency bin in the block of audio. We use a block size of 4096 and a 50% overlap window in our blocks (hop size = 0.5*block size). This section will now provide a short description of each feature with its mathematical representation.

1) Root Mean Square: (also known as the Quadratic Mean) is a very standard approach for measuring the average loudness of a signal over time.

$$M_x(\beta, n) = \sqrt[\beta]{\frac{1}{\mathcal{K}} \sum_{i=i_s(n)}^{i_e(n)} x^\beta(i).}$$

Fig. 1. $\beta = 2$

2) Zero Crossing Rate: measures the number of times per block that the signal changes sign in consecutive samples. This can be visualized simply as crossing the x-axis. This is a particularly meaningful signal for distinguishing silence (in which the ZCR will be near 0) from noise or music.

$$v_{\mathrm{ZC}}(n) = \frac{1}{2 \cdot \mathcal{K}} \sum_{i=i_s(n)}^{i_e(n)} \left| \operatorname{sign}\left[x(i)\right] - \operatorname{sign}\left[x(i-1)\right] \right|$$

3) Spectral Centroid: represents the Center of Gravity of the spectral data.

$$v_{\mathrm{SC}}(n) = \frac{\sum_{k=0}^{\mathcal{K}/2-1} k \cdot |X(k,n)|^2}{\sum_{k=0}^{\mathcal{K}/2-1} |X(k,n)|^2}$$

4) Spectral Rolloff: measures the frequency bin in which the cumulative magnitudes reach a predetermined percentage of the overall magnitude total. We chose that parameter (kappa) to be 85% as this is a commonly used standard.

$$v_{\mathrm{SR}}(n) = i \left| \sum_{k=0}^{i} |X(k,n)| = \kappa \cdot \sum_{k=0}^{\kappa_{/2-1}} |X(k,n)| \right.$$

Fig. 2. $\kappa = 0.85$

5) Spectral Crest Factor: represents a basic indicator of how tonal the signal is. It's calculation divides the maximum frequency bin magnitude by the sum of magnitudes.

$$v_{\mathrm{Tsc}}(n) = \frac{\max\limits_{0 \le k \le \mathcal{K}/2-1} |X(k,n)|}{\sum\limits_{k=0}^{\mathcal{K}/2-1} |X(k,n)|}$$

6) Spectral Flux: measures the difference in spectral magnitudes between blocks.

$$v_{\mathrm{SF}}(n) = \frac{\sqrt{\sum\limits_{k=0}^{\mathcal{K}/2-1} \left(|X(k,n)| - |X(k,n-1)|\right)^2}}{\mathcal{K}/2}$$

7) Mel Frequency Cepstral Coefficients: represents the spectral envelope of the audio block. This feature requires a certain number of filters in the calculation, and we elected to use the standard of 24.

$$v^j_{\mathrm{MFCC}}(n) = \sum_{k'=1}^{\mathcal{K}'} \log\left(|X'(k',n)|\right) \cdot \cos\left(j \cdot \left(k' - \frac{1}{2}\right) \frac{\pi}{\mathcal{K}'}\right)$$

This concludes the features. After extracting all seven features, we then calculate the mean and standard deviations of each feature for one second increments instead of the previous block size of 4096 samples. This greatly aggregates the data and is the last step in preparing the feature data for the SVM.

We also need a boolean array for each file to inform the SVM as to which samples of the feature data represent music and which do not. As mentioned in the introduction, we already had txt file annotations generated manually for around 1500 of the auditions. We used these to generate ground truth arrays for all of the files we use for training and testing.

For the actual training of the SVM we used the sklearn package and it was a straightforward implementation. The feature data was formatted in a matrix so that the SVM would see all 62 dimensions (7 features + MFCC's 24 filters; mean and standard deviation) for each sample, and the ground truth array. This generated the classifier fit, which is able to generate prediction arrays for new audition data with the same feature data formatting.

We only perform one treatment of the prediction array with a smoothing process. A simple example of this is demonstrated below.

0	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	0	0	1	0	0	0	0	0	0

0 = non music, 1 = music

In this example, this chart shows that at the 6 second mark, music was predicted by the classifier for this one second block. This is highly unlikely due to the fact that the exercises are longer than one second, so it is safe to assume that this is a prediction error. From this assumption we elect to switch this prediction value at index 6 to a zero. Our algorithm flips any index that is surrounded on both sides by non-matching states. The output of the example would be as shown below.

0	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	0	0	0	0	0	0	0	0	0

This process is also done for the positive case as well - if a non music value is predicted in between two predicted music values.

III. Evaluation

Our evaluation consists of a calculation of seven metrics: Positive Predictive Value, Negative Predictive Value, Sensitivity, Specificity, F1 Score, F1 Score Inverted, and Balanced Accuracy. We use the inverted versions of these metrics as well because they can provide further insight into the data that is not captured solely by the standard metric calculations.

1) Positive and Negative Predictive Value: the Positive Predictive Value (PPV) is also called Precision. Precision measures how many positives were true out of the number of positives predicted. The Negative Predictive value is achieved by inverting the prediction array and ground truth array logically and performing the same operation.

$$Precision = PPV = \frac{TP}{TP + FP} \tag{1}$$

$$Negative\ Predictive\ Value = NPV = \frac{TN}{TN + FN} \tag{2}$$

2) Sensitivity and Specificity: represent the True Positive Rate (TPR or Recall) and True Negative Rate (TNR). Recall measures how many positives were true out of the total number of exact positives (meaning the number of positives from the ground truth vector).

$$Sensitivity = TPR = \frac{TP}{TP + FN} \tag{3}$$

$$Specificity = TNR = \frac{TN}{TN + FP} \tag{4}$$

3) F1 Score: (or F Measure) is the harmonic mean of Precision and Sensitivity.

$$F1\ Score = \frac{2}{\frac{1}{PPV} + \frac{1}{TPR}} \tag{5}$$

$$F1\ Score_{Inv} = \frac{2}{\frac{1}{NPV} + \frac{1}{TNR}} \tag{6}$$

4) Balanced Accuracy: Balanced accuracy finds the average of the accuracy for true positives out of exact postives, and the true negatives out of exact negatives.

$$Accuracy = \frac{TPR + TNR}{2} \tag{7}$$

IV. Results

We calculate all of the metrics above for each file and then average them to get seven overall result values.

1) Test 1: These tests were done to measure the benefit of the smoothing process of the prediction arrays. We used 70 files for training from the 2018 concert band auditions, 35 of these were Flute auditions, and 35 were Bb Clarinet auditions. Then we tested on 30 auditions from the same year and band, 15 were Flutes, and 15 were Bb Clarinets.

	Without Smoothing	With Smoothing
PPV	0.97744	0.97640
NPV	0.87381	0.91951
TPR	0.97963	0.98749
TNR	0.85330	0.84639
F1	0.97828	0.98168
F1Inv	0.97828	0.87071
BAcc	0.92808	0.92922

This indicates some marginal performance improvements for the results With Smoothing. One noteable change is the slight increase in Sensitivity (yellow) and slight decreases in Specificity (green). There is also a significant decrease in the inverted F measure score.

In general, these results are very good, although the classifier does tend to be more Sensitive than Specific.

2) Test 2: This is the primary test of our classifier. We use all of 2017 data, all three bands (Middle School, Concert band, and Symphonic band), and all three instruments we have (Alto Saxophone, Bb Clarinet, and Flute). This were 705 audition files, but far more feature data samples. Each audition is around 5 to 6 minutes in length, and each second of audio data serves as one feature data sample. That means that for each audition, we have an average of 330 new feature data points for training the SVM. We tested on all 2018 Concert Band Flutes (145 files, with and without smoothing), all 2018 Concert Band Bb Clarinets (121 files, with and without smoothing), and then repeated Test 1 with smoothing.

	BbClar	BbClarWS	Flute	FluteWS	Test1WS
PPV	0.98616	0.98959	0.98561	0.98814	0.97596
NPV	0.83010	0.88035	0.87979	0.91023	0.91510
TPR	0.97385	0.98166	0.97968	0.98493	0.98595
TNR	0.90363	0.92906	0.90121	0.91869	0.84449
F1	0.97981	0.98545	0.98240	0.98633	0.98059
F1Inv	0.85965	0.89813	0.88056	0.90585	0.86158
BAcc	0.94577	0.96073	0.94787	0.95796	0.92789

WS = With Smoothing; Precision (Cyan), Sensitivity (Yellow), Specificity (Green)

These results from the larger tests are really positive. The precision is marginally increased across the board, the sensitivity is maintained, and the specificity is significantly increased. This data shows a much better classification in that its sensitivity and specificity are both improved, and a little better balanced (especially in the tests with smoothing). There is an average difference between sensitivity and specificity for the new test results (not including Test1WS) of 0.0669, which greatly outperforms the 0.1337 average difference from Test 1. Test1WS is excluded in that averaging calculation because this measure shows the performance on a different year and the difference that it makes. To implement this classifier, it will need to be trained on past years, and tested on the current year of its use going forward. The results from repeating Test 1 WS are comparable but slightly worse than the actual Test 1 results WS in every category. This could be because the training auditions were playing the same lyrical exercises, technical exercises, and sight reading examples as the testing auditions in Test 1, whereas they are all different in Test 2 because it is a different year.

V. Conclusion

Our goal was to create an algorithm that could automatically segment the audio data of FBA All-State band auditions by the classification of music vs. non music segments. Our approach is not particularly novel, this feature extraction and SVM classifier combination has been done in many other music informatics applications. But our specific choice of features is very effective and even recommendable in this specific use case or similar applications. However there is one additional step that is required for this problem to be adequately solved. Because this is just one step in the larger puzzle of actually predicting the scores for these auditions, our system needs to output a txt file with the same formatting as we had been doing manually. This is important because we know that there are exactly five parts to each audition: lyrical exercise, technical exercise, chromatic scale, 12 major scales, and sight reading. In the analysis of our current system, it does a very good job of predicting music vs. non music but there is going to need to be an additional step of logic and treatment of these predictions to yield the correct segmentation that will be needed to serve its purpose in the larger picture. The smoothing process is a step in the right direction, because it prevents any unlikely classifications of non music that may occur during an exercise section. But much more treatment will be needed to ensure only five succinct sections are recorded along with accounting for any uncommon or special cases.

References

[1] A. Lerch, *An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics*, 1st ed. The Institute of Electrical and Electronics Engineers, Inc, 2012.

[2] D. Olson, *Advanced Data Mining Techniques*, Springer-Verlag Berlin Heidelberg, 2008.

[3] D. Altman, *Diagnostic Tests 1: Sensitivity and Specificity*, BMJ, 1994.

[4] I. Guyon and A. Elisseeff, *An Introduction to Variable and Feature Selection*, Journal of Machine Learning Research, vol. 3, pp. 1157-1182, 2003.

[5] G. Guo and S. Z. Li, *Content-Based Audio Classification and Retrieval by Support Vector Machines*, Transactions on Neural Networks, vol. 14, no. 1, pp. 209-215, 2003