Clustering of Music/Nonmusic Classifications for ISMIR 2021

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

***Abstract*—The topic of this paper is the post-processing of binary classifications into a known number of sections. We implement a simple median filter as well as an informative process for removing excess sections in files that do not fit the section number requirement.**

I. Motivation

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 network is designed to evaluate the music sections individually, 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 solving this problem, we can greatly increase the dataset for the neural network, and in doing so, improve the results of the network's ability to learn the features that best predict the scores.

II. Context

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. The feature data is then written to a file which can be referenced without the need to recalculate all of the features each time. We use these feature data files to train a Support Vector Machine model for classifying future inputs. The model is trained on the instruments Bb Clarinet, Flute, and Alto Saxophone for all bands in 2017. The tests are performed on Bb Clarinet, Flute, and Alto Saxophone in Concert band of 2018. For this portion of the project specifically, we are now concerned with the post-processing of test prediction binary arrays to fit the data into the correct number of exercises. Because these files come from All-State auditions and we have the music that the students are playing, we want to force our predictions into the correct number of music sections.

III. Method

Our approach involves 3 primary processes: (1) median filter of raw binary classification predictions, (2) removing the shortest segment until the desired number of sections is reached, and (3) an informative process using the best predicted files to inform which segments to remove in other files.

1) Median Filter: The median filter is implemented to smooth the data. The goal of this step is to make sure that there are no 1-second sections. This is reasonable because in a valid performance file, no musical section or pause between exercises would realistically be one second, and it is more likely an error in the prediction. By flipping these isolated occurrences, we reduce excess sections and help the predictions to include more contiguous sections. An example in which the median filter would fix an error is shown 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

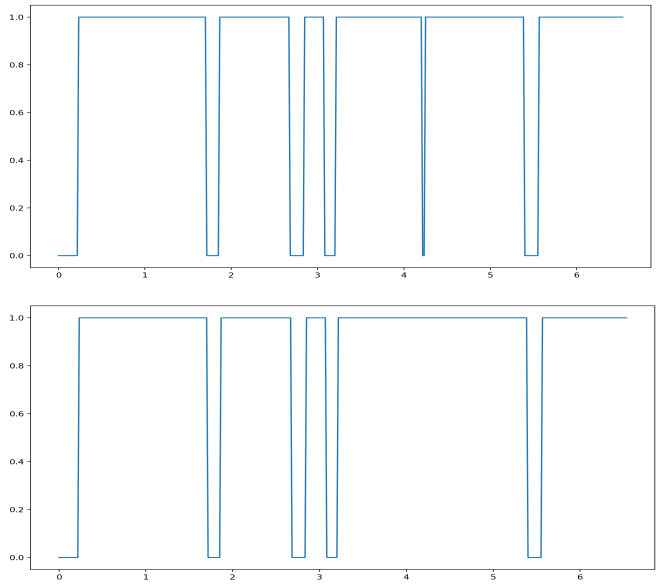
In this example, this chart shows that at the 6 second mark, music was predicted by the classifier for this one second block. This is highly unlikely due to the fact that the exercises are longer than one second, so it is safe to assume that this is a prediction error. From this assumption we elect to 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 short nonmusic case as well: if a nonmusic value is predicted in between two predicted music values. We implement this median filter in the order as described, first the outlier music cases, and then the outlier nonmusic ([0 1 0] before [1 0 1]). There is no particular logic involved that favored one over the other, we measured better test results with this order, rather than the inverse. Additionally, the potential for errors is present regardless of which order is chosen, and the following steps will ideally catch any errors made at this step.

2) Removing Shortest Excess Section: The next step is removing the shortest sections until the desired number is

reached. The reasoning behind this approach is that in our observations of the files, small isolated 2 or 3-second error sections (longer than 1 second and therefore being skipped by the median filter) would occur within a musical section. By systematically and iteratively flipping the shortest segment each time until the correct number of sections is reached, we always force the prediction into the correct number of groups, and rely on the accuracy of our predictions overall to make these flips do more good than harm in the prediction accuracy.



0.0 = nonmusic, 1.0 = music

Fig. 1. Example of flipping shortest segment

In this example you can see the result of flipping the shortest segment to produce a more accurate prediction. These two plots are of the same audition file prediction. The top one has one section too many, you can tell because each audition only has 5 musical sections, and this one has 6. Samples predicted as music are a 1.0 and samples predicted as nonmusic are a 0. It is clear in this case that in the middle of the fourth exercise, there should not be a two second long section of nonmusic, but because it is two seconds long it was skipped over by the previously described median filter. Therefore, the shortest segment is flipped which gives a correct prediction for this file with 5 separately identified musical exercises.

We do have a small variation in this step which we found fits the data better and we deemed it to not be excessively overfitting. We found a number of files had an isolated 2-3 seconds of music predicted before the clearly visible beginning of the first exercise. We could not simply prioritize flipping short music segments over flipping nonmusic segments because then there would frequently be parts of exercises cut off due to incorrect nonmusic predictions near the end of an exercise. Our solution to this problem was to flip the music segment only if it was less than 4 seconds, and there are less than 6 predicted music samples in the surrounding range of 8 samples before and 8 samples after the selected segment. For example, the first example below would be flipped, but the second would not due to the number of surrounding samples predicted as music. The first demonstrates the isolated music section example we want to remove, the second shows how this avoids flipping segments at the end of a music section. These specific window sizes were selected from comparison of results and trial and error.

0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0	0	0	0	0	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

0 = non music, 1 = music
Flipped

0	1	2	3	4	5	6	7	8	9	10	11	12	13
1	1	1	1	1	1	1	1	1	1	0	0	1	1
1	1	1	1	1	1	1	1	1	1	0	0	1	1

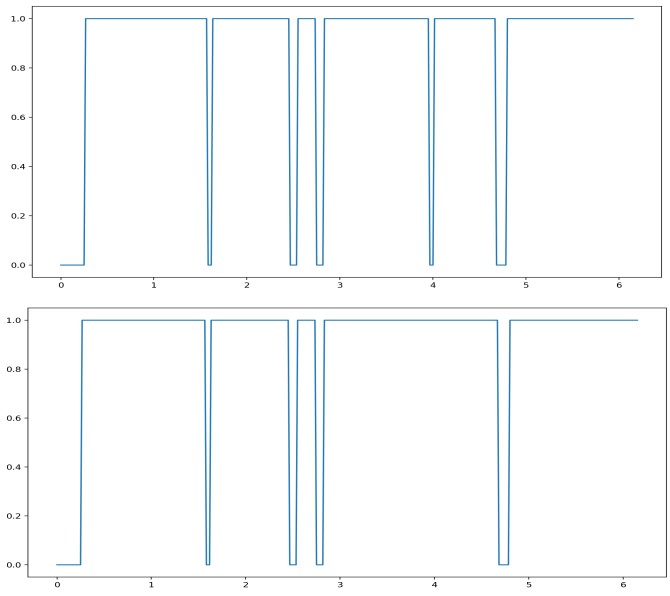
0 = non music, 1 = music
Not Flipped

3) Informative Process: Although the combination of steps (1) and (2) is fairly robust, there is still is too high a likelihood of an error occurring and the wrong section getting flipped. The reason for this is that most of the time the short segments are mistakes, but sometimes there will be a tie of two segments for the shortest. In this case, the algorithm from step (2) was designed to simply choose the first occurring segment to flip, which is basically a 50% chance of being correct. We needed a way to inform this last flip where the error is most likely to occur. The approach we use for this informative process has a few steps:

1) Apply smoothing to all files
   - If file has correct number of segments, its a good file and we save its music and nonmusic segment lengths
   - If it has more than the correct number of segments, store its prediction data (prevents recalculating) and label it as bad
   - If it has less than the correct number of segments, flag it to be manually annotated
2) Calculate mean and standard deviation for each music and nonmusic segment lengths from good files
3) Iterate through and process each bad file
   - Perform Shortest Segment algorithm from step (2) until the last flip
   - Flip each segment option iteratively, and measure the probability of the segments compared to the good file distributions for each flip option

- For each flip option, multiply the probabilities of each segment for a total product
- Determine which segment flip provides the highest overall probability (closest to distribution of good files)

4) Flip the selected segment



0.0 = nonmusic, 1.0 = music

Fig. 2. Example of informed process

This example shows the necessity of the informed process. In this example there is a tie between the nonmusic segment around 1 minute and 40 seconds, and the nonmusic segment at 4 minutes. Both segments are 2 seconds long, and the shortest segment algorithm from step (2) would choose the first one to flip. However, in this case, that would be incorrect and would combine the first and second exercises while splitting the fourth in half. With the informed process however, the algorithm can tell that flipping the 9th segment (it tries flipping each one, music and nonmusic) results in a prediction that best matches the good file distribution for this testing set - and flips the correct segment.

IV. Evaluation

A. Classification Metrics

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. For all following calculations True Positives (TP) are the number of samples correctly predicted as Music, True Negatives (TN) are the number of samples correctly predicted as Nonmusic, False Positives (FP) are the number of samples incorrectly predicted as Music, and False Negatives (FN) are the number of samples incorrectly predicted as Nonmusic.

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.

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

B. Stamp Metrics

Because we truly care about the segment boundaries for each exercise, we also began to incorporate the time stamp deviations for each musical segment's start and end point against the ground truth in our evaluation. We measure:

- Mean Distance: The seconds of difference between the ground truth boundary timestamps and the predicted boundary timestamps (negative means early predicted boundary, positive means late predicted boundary).
- Mean Absolute Distance: The absolute value of the seconds of difference between the ground truth boundary timestamps and the predicted boundary timestamps.
- Standard Deviation of Distance: The standard deviation of the difference (not absolute distance) between the ground truth boundary timestamps and the predicted boundary timestamps.

V. Results

We calculate all of the metrics above for each file and then average them to get seven overall result values, and averaged stamp deviation data for each test set. We will compare the results of the two primary approaches we attempted, Smoothing and Shortest Segment Removal - method steps (1) and (2), and Smoothing and Informative Process - method steps (1) and (3).

A. Prediction Metrics

The table below shows the prediction metrics of the shortest segment removal algorithm compared to the informative process for all three instruments (Bb Clar, Flute, and AltoSax).

	B-R	B-I	F-R	F-I	A-R	A-I
PPV	0.99	0.99	0.99	0.99	0.99	0.99
NPV	0.91	0.90	0.92	0.90	0.92	0.87
TPR	0.99	0.99	0.99	0.99	0.99	0.97
TNR	0.95	0.94	0.93	0.94	0.95	0.96
F1	0.99	0.99	0.99	0.99	0.99	0.98
F1Inv	0.92	0.92	0.92	0.92	0.93	0.90
BAcc	0.97	0.97	0.97	0.97	0.97	0.97
	BbClar		Flute		AltoSax	

R = Remove short segments only, I = Informative Process;

These results are not significantly different for BbClar and Flute, but the AltoSax has significant decreases in NPV and F1Inv. This may seem disappointing, but we have found that because our project goal is centered around the timestamp locations, the stamp metrics better evaluate our success.

B. Stamp Metrics

	MeanDist	AbsMeanDist	StdDev
BbClar R	1.45	2.39	10.53
BbClar I	-0.38	1.04	2.36

R = Remove short segments only, I = Informative Process; Bb Clarinet

	MeanDist	AbsMeanDist	StdDev
Flute R	0.45	1.39	6.22
Flute I	0.25	0.98	3.3

R = Remove short segments only, I = Informative Process; Flute

	MeanDist	AbsMeanDist	StdDev
AltoSax R	2.30	3.11	11.29
AltoSax I	3.65	4.06	16.43

R = Remove short segments only, I = Informative Process; Alto Saxophone

Here we see that the stamp results improve greatly for BbClar and Flute. But similar to the prediction metrics, Alto Saxophone gets worse. This shows that the algorithm generally works better as a solution to the problem, but that there seems to be something in the Alto Saxophone data that is causing it to perform worse than BbClar and Flute.

VI. Conclusion

Our goal was to process the predictions from our SVM model such that they fit the required number of sections, and are as accurate as possible. Our approach is not particularly novel, although there is something to be said regarding the strange aspects of this problem as both a classification and a clustering problem. Our approach is performing at a level of accuracy that is actually functional for the BbClar and Flute. Going forward if we can figure out and fix whatever is causing the lower performance in AltoSax then we will have a usable system. Because this is just one step in the larger puzzle of actually predicting the scores for these auditions, our system needs to output a text file with the beginning and end time stamps for each musical exercise. It also needs to be able to flag outliers that will require manual annotation. We can already flag some outliers as mentioned earlier for having less than the desired number of sections after the smoothing process. The text file output and outlier flagging are not going to be difficult tasks once we get the process fully functional with AltoSax as well.

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