Instrument Recognition in African Music Field Recordings

Shawn Anderson
Computing Science
University of Alberta
swanders@ualberta.ca

Maxime Convert
Computing Science
University of Alberta
convert@ualberta.ca

Tarek El-Bohtimy
Computing Science
University of Alberta
elbohtim@ualberta.ca

Lingbo Tang
Computing Science
University of Alberta
lingbo@ualberta.ca

Yufei Zhang
University of Alberta
Computing Science
yufei2@ualberta.ca

Coaches and Clients:

Dr. Michael Frishkopf
michaelf@ualberta.ca

Dr. Russell Greiner
rgreiner@ualberta.ca

Dr. Abram Hindle
abram.hindle@ualberta.ca

Abstract

The archiving of music field recordings is often a tedious and time-consuming task, sometimes requiring the listening of hundreds of audio files. Just as any repetitive task, the development of an automated classification system could greatly help music researchers tackle such a problem more efficiently. This project is an attempt at creating a machine model recognizing the presence of certain instruments within audio files, by using music information retrieval techniques. The focus of the study was on classifying African music field recordings, and in particular, three specific sounds were chosen as the basis for the experiments: vocals, drums and handclaps. The problem was addressed by experimenting and retrieving a specific set of sound signal features (using the feature extraction tool MARSYAS) from multiple audio datasets, and then comparing the results returned by different classification algorithms. Overall, the K Nearest Neighbor (KNN) classification algorithm showed to be better than Support Vector Machine (SVM). Further, retrieving features over short overlapping segments of a music track proved to be more successful than non-overlapping segments and tracks with extended length. However, further investigations remain to be conducted to achieve a convincing machine timbre recognition system.
The code is available at: https://github.com/LingboTang/466Team7Music

1 Introduction

1.1 The Problem

Dr. Michael Frishkopf, an ethnomusicology researcher at the University of Alberta has personally recorded a large number of field recordings of African music. Dr. Frishkopf requires, for his research, that each track be labeled with which instruments are present. Manual annotation of the songs is a laborious and time consuming task; thus, it would be greatly beneficial to have an automated system that can label tracks with the instruments they contain. Labeling music with which instruments are present is a generalizable problem, a solution which may be useful to researchers and practitioners in the future.

The applications of automated instrument recognition span beyond the scope of this problem. Consider audio recordings in general, recordings of speeches, conferences, concerts, lectures and more, in each case, a recording is often monolithic and lengthy, and requires manual searching to find specific points that a user may be interested in. An automated segment classifier would allow a user to query such an audio file and automatically be given segments which contain what they are looking for, saving the user the need to manually listen through undesired portions. Another practical application is the synergistic potential to be used with search engine technologies. Consider the ability to annotate any audio that exists on the Internet, a search engine could then search among annotations. For example, a user could query "'1980's' 'sitar' 'theremin'" and be given audio clips that have been labeled as such. The wide range of possible applications motivated us to investigate further on this problem.

We began our investigations on the specific problem of classifying 30 second tracks of African music, focusing on only three instruments to begin with: claps, vocals, and drums. We chose these three because they were the most common labels in the dataset. The task was to label tracks with which instruments they contain, and more precisely, the exact segments which contain particular instruments. The first step in applying Machine Learning to such a problem was the ability to extract features from audio, secondly, experimenting with various algorithms and parameters which maximize the correctness of the classification based on the extracted features.

1.2 Related Work

Music Information Retrieval (MIR) is an emerging field with notable successful applications such as Shazam. Shazam is an application which allows users to discover the title and artist of a song from recording a 15 seconds clip of that song. The clip is encoded and sent to Shazam servers and analyzed by it's spectrogram.[4] The work done by the team at Shazam is a display of the powerful utilization of music information retrieval. Shazam only uses a single feature of music in their analysis; intuitively, using more features has the potential of yielding even greater results, thus, the inspiration for applying machine learning techniques on an extended set of features.

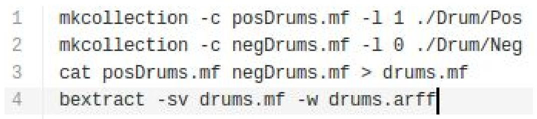
2 The Environment setup

2.1 The Data

The training data was obtained from the Smithsonian Folkways website[9], from which we downloaded all 500 free sample clips from their International library of African Music (ILAM) collection. Each clip is exactly 30 seconds in length and is labeled with which instruments it contains. As mentioned in the previous section, we focused on vocals, drums and handclaps.

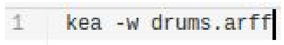
2.2 The Tools

Marsyas (Music Analysis, Retrieval and Synthesis for Audio Signals) is an open source software framework for audio processing with specific emphasis on Music Information Retrieval applications. It has been designed and written by George Tzanetakis with help from students and researchers from around the world.[1] We used Marsyas to extract audio features that can be used for machine learning. To use Marsyas to create a feature file, first create a Marsyas collection, appending the appropriate labels to each example. Then, use bextract binary file to create a .arff file which contains feature values:



```
mkcollection -c posDrums.mf -l 1 ./Drum/Pos
mkcollection -c negDrums.mf -l 0 ./Drum/Neg
cat posDrums.mf negDrums.mf > drums.mf
bextract -sv drums.mf -w drums.arff
```

Figure 1-a: Feature Extraction with Marsyas



```
kea -w drums.arff
```

Figure 1-b: Classification with kea

Marsyas has a built-in machine learning tool called Kea (derived from Weka) which can be used to apply classification algorithms on the .arff file. Learning a classifier by SVM with linear kernel and 10-fold cross validation can be achieved in Marsyas with the command in Figure 1-b.
A .arff file can also be used directly with Weka which has many options for visualizing data and experimenting with different classifiers.

2.3 The Features

Zero-Crossing

Where the sign of the spectrum changes. [1]

Spectral Centroid

Indicates where the “center of mass” of the spectrum is, and is calculated by the following formula:

$$C = \frac{\sum_i i A_i}{\sum_i A_i} \quad (1)$$

where A is the amplitude, and i is the bin number.[1]

Spectral Flux

The euclidean distance of the difference between the magnitude of the short time Fourier transform spectrum evaluated at two successive sound frames. This measures how quickly the spectrum is changing. [1]

Spectral Roll-off

A roll-off point is the frequency in which N% of the magnitude distribution is concentrated (N is usually 95). [2]

Mel-Frequency Cepstral Coefficients (MFCC)

Sounds generated by a human mouth are filtered by the shape of the vocal tract (such as the position of the tongue, or the shape of the lips). This shape determines what sound comes out of the mouth. If we can determine the shape accurately we can get an accurate representation of the phoneme being produced. The vocal tract manifests itself in an envelope and MFCC accurately represents this envelope. Computed by grouping the short time Fourier transform points of each frame into a set of coefficients by a set of weighting curves with logarithmic and discrete cosine transform. [2]-[3]

Chroma

Chroma is an attribute of pitch. Pitches contained in the same pitch class have the same chroma.
A pitch class is set of pitches that are a whole number of octaves apart, for example, all the C notes on a piano are contained in the same pitch class thus share the same chroma.

In our study, we are taking the average values of the features described above over segments of sounds. Therefore, the features we used to train classifiers correspond to statistics of each features, i.e. the average and the standard deviation. This resulted into a total of 124 features.

3 The Experiments

3.1 Average feature values over 30-seconds tracks, with binary classifiers

The first step was to collect all 500 tracks and their corresponding labels. We transformed all the tracks to WAV format to be compatible with Marsyas, and wrote a script to sort all the tracks into folders with their corresponding labels. For three instruments options there are eight combinations, thus the files were sorted into eight folders. We chose to begin experimenting with binary classifiers for each respective instrument instead of a multi-class classifier. This required sorting the data further into positive and negative instances for each respective instrument. Once this was done we were able to extract features using Marsyas, an example of extracting features for a drums classifier can be seen in Figure 1-a.

Marsyas extracts features 86 times per second and averages features over an entire audio clip. For this reason, our team came to the conclusion that extracting features over an entire 30 second length clip will not be sufficient for accurate features. For example, a 30 second track containing claps will have less than one second of total aggregate clap time and the features identifying with claps will get washed out from averaging over 30 seconds of other noise. We concluded that a next iteration experiment will require shorter track length.

Using Marsyas' integrated machine learning, we used our initial data to train a classifier using SVM with a linear kernel and testing using 10-fold cross-validation, demonstrated by Figure 1-b. We further decided to train another classifier, KNN, to get a comparison between two different algorithms. We chose K=11 (11-NN), by following the rule of thumb of taking the square root of the number of features used for training (√124 in our case). The excitement of seeing 98% accuracy for classifying vocals quickly faded as it became apparent how skewed our data was. Our vocal dataset was 98% positive, resulting in a naive classifier that simply labels all tracks with vocals. These results revealed the necessity of a more sophisticated measure of success than just accuracy. We began using F1-measure to quantify the success of our models, F1 was sufficient because there is not a significant cost difference between false positives and false negatives. The initial results suggested also that we must have a more balanced dataset for future iterations of experimentation.

$$F1 = \frac{2TP}{2TP + FP + FN} \quad (2)$$

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \quad (3)$$

3.2 Average features over 1-second segments, with binary classifiers

To address the issues mentioned above we chose to split the 30-seconds tracks into 1-second chunks and manually label each second with only the instruments contained in that second (which took approximately 15 manual hours). The new dataset contained approximately 15,000 tracks with less variance and had the advantage of having a more balanced dataset for vocals.

With our new dataset we expected to have a strict increase in F1-measure among all instruments. With vocals this was certainly the case. Among claps and drums however, there was a decrease in F1-measure, with drums having only slightly decreased, whereas claps had a significant decrease. This result came as a surprise to us and presented inspiration for further investigation. The focus was now on claps and how we could improve our results. It became apparent that perhaps in the process of systematically chopping tracks by seconds we may have chopped the audio in the midst of a clap, resulting in a partial clap on one chunk and a partial clap on another chunk.

3.3 Average feature values over 1-second overlapping segments, with binary classifiers

If segmenting claps occurred often enough it would give a mis-representation of the features. We decided to further expand our dataset into overlapping chunks rather than disjoint chunks. Doing so would eliminate the problem while also further reduce the skewness of the data and provide more training data. Each new overlapping segment consists of 50% of the 1-sec chunk preceding and 50% of the subsequent chunk. Again, these data were manually labeled, which took around 15 more hours of manual time. This resulted in a new dataset of approximately 30,000 overlapping segments, on which we applied the two algorithms for the next iteration of our experiment.

3.4 Average feature values over 1-second overlapping segments, with multi-class classifiers

As a final experiment we chose to train a multiclass classifier using our overlapping data. There were 8 classes, representing all combinations of the three labels. With this model we expected a high error rate, as the mistake of classifying a chunk that contains vocals and claps as a chunk that contains vocals and drums seemed very likely, along with other similar cases of this sort.

4 Results and Discussion

4.1 Vocals

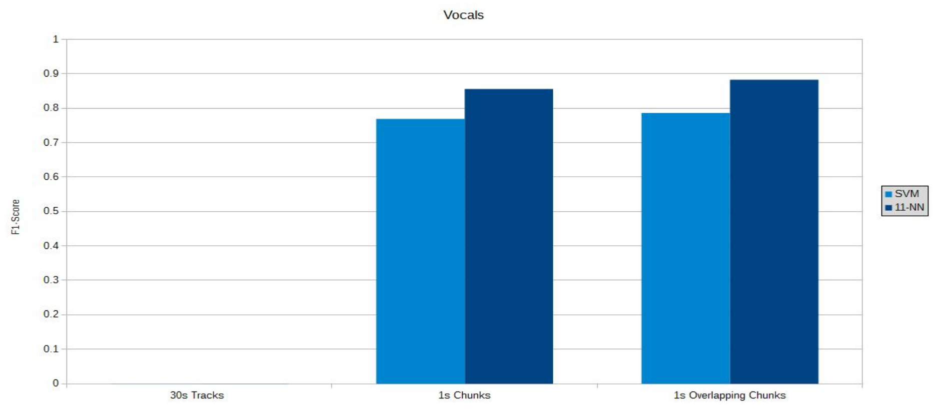


Figure 4-a. F1-Score of vocals using different samples and algorithms

Since vocal data is heavily skewed, the classifier adopts a majority class policy and classifies all tracks as having vocals, which results in an F1-score of 0 (see figure 4-a.). By creating 1-sec chunks and overlapping chunks from the original data, we obtained more segments not containing vocals. As the number of negative samples increases, the data is more balanced; hence the F1-Score increased to more than 0.8 for 11-NN, and we see that 11-NN performs better than SVM.

Comparing the results for both algorithms between non-overlapping chunks and overlapping chunks, we observe a slight increase in performance for the latter overlapping set.

4.2 Drums

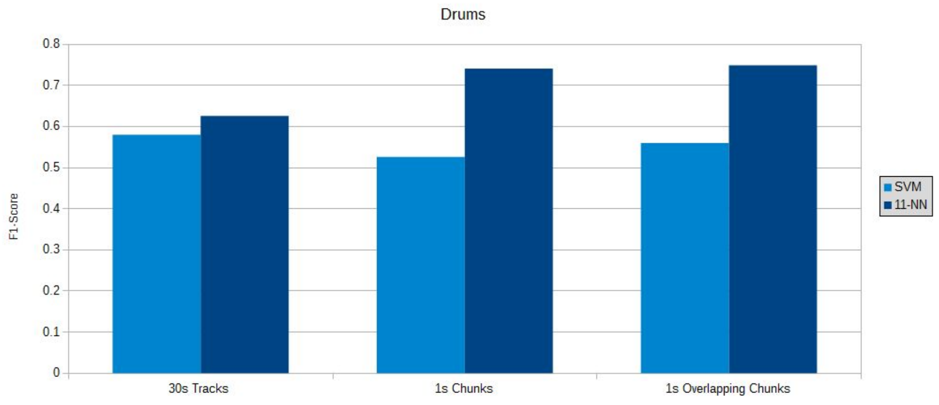


Figure 4-b. F1-Score of drums using different samples and algorithms

Compared to vocals, drums were not as present in the 30-sec tracks. Hence the data is less skewed, giving comparable performance across the first and the last two experiments. Looking at the bar-plots, we again observe that KNN outperforms SVM for each case. The first experiment gave a poor performance of around 0.6 for KNN. This is due to the feature values specific to drums being washed out by the other sounds in the whole track. We then observe an improvement for the 1-sec segments in the last two experiments, which stands at around 0.7. SVM slightly decreases performance from the 30-sec tracks to the 1-sec chunks in the last two experiments.

4.3 Claps

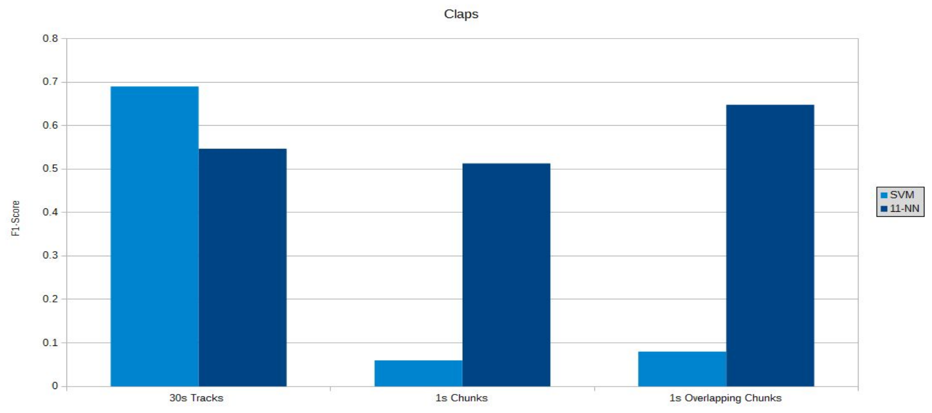


Figure 4-c. F1-Score of claps using different samples and algorithms

SVM did well for 30-sec tracks but poorly for 1-sec chunks and overlapping chunks. During its training phase, SVM uses hinge loss to balance the weight on more relative data and less relative data. Since its consideration mostly depends on support vectors, it cannot reduce the weight of the data which is farther from the hyperplane cluster than we expected. When we split the data into 1-sec chunks, we unwillingly generated more data that is irrelevant to the hyperplane which weights will be added into the false positive and false negative classifications.

The classification using KNN still gave us reasonably convincing F1-scores, which again reveals that KNN is better on small relevant datasets. Regardless of the amount of the data, KNN simply calculated the distance and will only rank the K-th nearest data, so the algorithm ignored most of the irrelevant data during the classification.

We still notice a decrease in performance for KNN between the initial experiment and the second experiment. This could be due to claps getting cut off between two chunks therefore not giving the algorithms a clear interpretation of what a clap is. Further, there were very few instances in which there were only claps so the algorithms had to deal with a lot of noise when identifying the specific sound of claps. As in the previous two instrument binary classifiers, we again observe a consistent improvement between the non-overlapping chunks in experiment 2 and the overlapping ones in experiment 3.

4.4 Multi-class

Overall 11-NN performed better than SVM in all cases. Identifying {Claps} and {Claps,Drums} there were few to no cases where those specific situations happened, therefore the learner did not have enough to train on, causing low results. When it came to recognizing {Drums} and {Vocals} the F1-scores are quite good, this is because in most cases there was usually no other instruments played at that time giving the learner a clear example of what a {Drum} or {Vocal} was. Also, when it came to recognizing that no instruments existed the learner did reasonably well. The problem occurred when it tried to classify two or more instruments at a time, this is because when there are vocals being played within a track the vocals are usually the prevailing sound therefore it is obstructing the other sounds causing the learner to not recognize if there are drums or claps in the track.

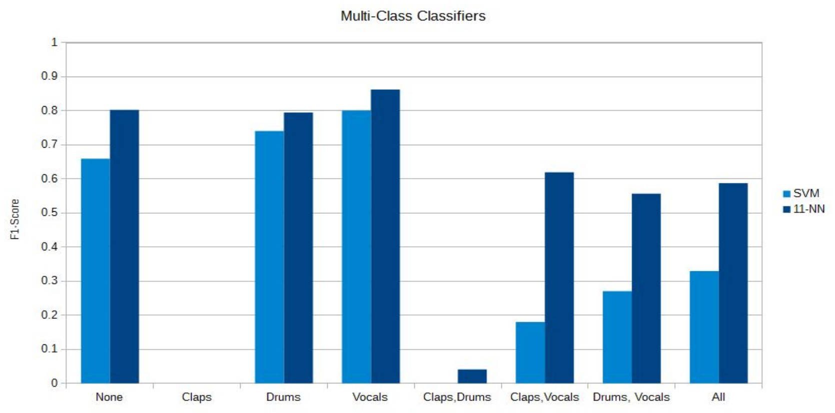


Figure 4-d. F1-Score of multi-class classifiers using different samples and algorithms

4.5 Result summary and example of the model usage

From the previously outlined result, we can say that the method that worked the best is the KNN classification algorithm (K=11), applied to the 30,000 overlapping chunks dataset. In order to visualize the result, below is a diagram showing how our classifier performed in recognizing instruments within a track:

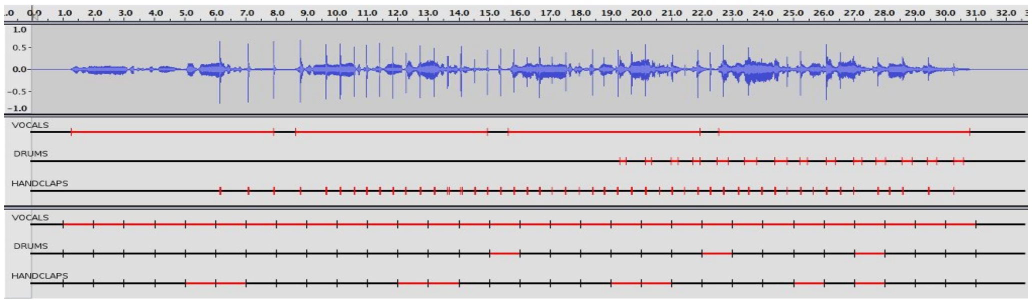


Figure 4-e: Ideal classification vs Classification of 11-NN on overlapping chunks [10]
(Song: Kob'a ntja, I.L.A.M. (1959) South Africa, Lesotho)

The timeline at the bottom represents how our classifier performed at classifying sounds on 1-sec chunks. The timeline right above it is how an ideal classification system should perform. As we can see, our classifier is still far from giving the desired result. However, by working toward obtaining increased performance on the sound classifiers, and working on tighter overlaps of shorter length, we could intuitively reach the ideal classification model.

5 Conclusion and Future Work

From the results of our experiments we can conclude that the 11-NN algorithm tended to perform better than SVM, and that chunking the data into 1-second segments and overlapping segments tended to increase our F1-scores. Overlapping the chunks did do better, but only slightly better in most cases. Binary classification performed better than multi-class classifiers in correctly recognizing whether or not vocals, drums, and claps are in a sound track.

The work done in this paper is still in its preliminary stages, and things that could be done to advance the results would be:

- testing with different sampling rates,
- ensuring the segments are small enough to not miss a clap or a drum sound,
- expanding to a wider range of instruments,
- using more features,
- using larger/more balanced datasets,
- applying to a general set of music genres.

We hope this study gives an intuition of what can be done, and what is to be done next in order to achieve machine timbre recognition.

6 References

[1] George Tzanetakis, Perry Cook (2000) MARSYAS: A framework for audio analysis, http://webhome.csc.uvic.ca/~gtzan/output/marsyas2000.pdf
[2]Xin Zhang,Zbigniew W. Ras(2007) Analysis of Sound Features for Music Timbre Recognition
[3] James Lyons(2012) Mel Frequency Cepstral Coefficient (MFCC) tutorial, http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/
[4] Avery Li-Chun Wang (2003) An Industrial-Strength Audio Search Algorithm, http://www.ee.columbia.edu/~dpwe/papers/Wang03-shazam.pdf
[5]Jayme Garcia, Arnal Barbedo, George Tzanetakis (2011) Musical Instrument Classification Using Individual Partials. In IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING, VOL. 19, NO. 1, JANUARY 2011, pp 111-122, http://webhome.csc.uvic.ca/~gtzan/output/taslp2010gtzan.pdf
[6]Ichiro Fujinaga (1988) Machine recognition of timbre using steady-state tone of acoustic musical instruments, http://www.music.mcgill.ca/~ich/research/icmc98/icmc98.timbre.pdf
[7]El Hachemi Alikacem, Houari A. Sahraoui (2006) Generic Metric Extraction Framework, https://www.crim.ca/Publications/2006/documents/plein_texte/ASD_AliE_IWSM06.pdf
[8]Daniel P. W. Ellis (2007) CLASSIFYING MUSIC AUDIO WITH TIMBRAL AND CHROMA FEATURES
[9]Smithsonian Folkways, © 2015 Smithsonian Institution, http://www.folkways.si.edu/
[10]Audacity Team (2008): Audacity (Version 2.1.0) [Computer program]. Retrieved Oct 7, 2015, from http://audacityteam.org/
[11]Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1