Music Genre Classification

Manuja Sharma, Shruti Misra
Department of Electrical Engineering University of Washington
Seattle, WA
manuja21@uw.edu, shrm145@uw.edu

Abstract

Automatic music genre classification is an active area of research within the field of Music Information Retrieval (MIR). It provides a framework for exploring and evaluating audio features which lend to a better understanding of content-based musical signals. In this project, we implement and analyze various machine learning techniques to classify audio tracks into 10 different musical genres. The main audio features used for this classification task are the timbral features, pitch content and loudness of an audio track. We use a modified subset of the Million Song Dataset (MSD) which provides already extracted feature vectors for 1 million songs. We first establish baseline models for the classification task such as Random Classification, Gaussian Mixture Models and Support Vector Machines. Next, we implement neural network models such as Convolutional Neural Networks (CNNs) and Convolutional Recurrent Neural Networks (CRNNs). Both the neural network models are able to surpass the accuracy of the baseline models and CRNNs perform better than CNNs. To understand the impact of different features on performance, we implement GMMs and CRNNs with just timbral and pitch features. Thus, we propose two neural network architectures that surpass the baseline models and are comparable to models found in existing literature. Moreover, in this report, we also conduct an analysis on the importance of different acoustic features that characterize musical genres

1 Introduction

Music genres are categorical labels used to characterize a piece of music. A music genre is defined by some set of common features shared by its members. The features are typically related to the rhythmic structure, harmonic information and the instrumentation of a musical piece. Pop, rock and classical are examples of popular music genres. The boundaries between different genres are not distinct as they arise through a complex interplay of historical, cultural, corporate and public factors. Therefore, classifying music into different genres is not as clear-cut since the definition of genres often tend to be subjective. Genre classification schemes mainly find their application in the field of Music Information Retrieval (MIR). This field lies at the intersection of musicology, signal processing and artificial intelligence. It is being employed by academics and businesses to categorize, manipulate and often create music. An example of genre classification in MIR is the music recommendation systems employed by companies like Spotify, Amazon Music, etc.

In this project, we focus on classifying music into ten different genres by using features extracted from the audio data derived from the Million Song Dataset (MSD). Our approach is two fold: (i) creating and evaluating baseline models such as random classification, Gaussian Mixture Models (GMMs) and Support Vector Machines (SVMs), and (ii) implementing and evaluating neural network models such as Convolutional Neural Nets (CNN) and Convolutional Recurrent Neural Networks (CRNN). For a dataset with 70% human accuracy, we achieved accuracy of 65% on a balanced test dataset. Neural nets performed better than the baseline models which had predictions in the range from 10 % to 50%. We also analyzed the importance of different features in classification and found

that timbre features provide better performance over chroma features with neural networks and GMM. Along with accuracy, we looked at the confusion matrix and F1 scores to evaluate different models. We have also discussed some future work that can help in improving the overall accuracy.

2 Related Work

Music genre classification has been a popular topic as summarized by Sturm (2012). For, this project, we have looked at previous work pertaining to MSD dataset and work with similar number of classifying labels. Dieleman et al. (2011), used the MSD dataset and classified music with an accuracy of 29.52% using convolution nets . They classified music into 20 different genres with a test set of 5000 songs. We have performed classification for 10 labels with a test set of 10,000 songs. Schindler et al. (2016), used shallow and deep CNN to perform genre classification on various datasets . Their model trained on 50,000 song from MSD with 15 classes has an accuracy of 67%, though the accuracy for other datasets like GTZAN and ISMIR is around 85%. We have achieved comparable accuracy to their MSD dataset using the CRNN model though we have used fewer class labels. Tzanetakis and Cook (2002) et al. used GMM for music genre classification and predicted 10 genre labels with 60% accuracy using a smaller dataset for both testing and training. They had a similar feature set consisting of timbre, pitch, and loudness though the total dimension of feature set for each segment was greater than the MSD dataset. The 10 class labels that they work with are slightly different than our genre labels. They include genres like classical music that has more distinct features for classification. More recent work, has looked at using recurrent neural networks along with CNN to perform genre classification. Choi et al. (2017) used CRNN with 2 layers of gated recurrent unit (GRU) to predict the top 50 tags on MSD based on genre, mood, instrument,etc. Their paper presents CRNN as a better alternative to CNN but doesn't provide any accuracy score specifically for genre classification.

3 Dataset

The Million Song Dataset (MSD) is a freely available compilation of audio features and metadata for 1,000,000 contemporary music tracks from 44,745 unique artists. In its original form, data for each track includes textual features such as artist and album names, numerical descriptors such as duration and audio features derived using a music analysis platform provided by The Echo Nest (since acquired by Spotify). The Million Song Dataset does not include any genre labels. However, external groups have proposed genre labels for some of the tracks by accessing external music tagging databases. For the genre classification task proposed in this project, the CD2C tagtraum genre annotations (Schreiber (2015)) are used. These annotations are derived from multiple source databases such as the beaTunes genre dataset, Last.fm dataset and Top-MAGD dataset. We use the CD2C variant with non-ambiguous annotations, that is, tracks with multiple genre labels are not included. As a result, we classify tracks into 10 different genres; Rap, Rock, RnB, Electronic, Metal, Blues, Pop, Jazz, Country, Reggae. For the purpose of this project, we use a subset of the Million Song Dataset, that consists of 40,000 training samples and 10,000 test samples The training and test sets are balanced, that is, there are equal number of samples for all classes. Additionally, the dataset we employ only consists of the audio features for each track and does not contain textual or numerical descriptors.

3.1 Features

The features for each audio track are partitioned into bins of 120 segments. A segment corresponds to an automatically identified, roughly continuous sections of audio with similar perceptual quality. In the modified version of MSD that we use, the number of segments per track is fixed to 120. For each segment, the following audio features are available: 12 timbre features, 12 chroma features and loudness at start of segment concatenated in that order. Consequently, each segment is characterized by a 25 dimensional vector. Furthermore, each audio track is represented by a total of 3,000 features (120x25 = 3000). Every feature has been normalized by subtracting the mean and dividing by the standard deviation.

3.1.1 Timbre Features:

The timbre features for each segment are given by 12 unbounded values roughly centered around 0. The 12 dimensional vector represents the Mel-frequency cepstral coefficients (MFCCs) for that segment. The mel-frequency cepstrum (MFC) represents the short term power spectrum of a sound. The MFC approximates the response of the human auditory system, by spacing the frequency bands equally on the mel scale. MFCCs are coefficients that collectively constitute an MFC. To extract the MFCC features, the Fourier transform of a signal window is computed. The power spectrum thus obtained is then mapped to the mel scale. Next, the log of the powers at each of the mel frequencies is computed. Finally, the discrete cosine transformation (DCT) of the mel log powers is evaluated. The MFCCs are the amplitudes of the obtained spectrum. In the Million Song Dataset, the MFCCs for each segment are computed using the Echonest API (Jehan and DesRoches (2011)).

3.1.2 Chroma Features

Chroma features represent the pitch content of each segment. These features are represented by a 12 dimensional vector (per segment), which corresponds to the 12 pitch classes C, C#, D to B, with values ranging from 0 to 1. The values describe the relative dominance of every pitch in the chromatic scale. The vector is normalized to 1, by their strongest dimension. Thus, noise is likely represented by values that are all close to 1, while pure tones are described by one value at 1 and others are near 0. The EchoNest API is used to extract the chroma features for each segment in MSD.

3.1.3 Loudness Feature

In the modified dataset, the loudness information is contained in a single value that represents the loudness at start of a segment. The timbral features also encode some loudness information, including maximum loudness. The loudness feature is in the units of decibels (dB).

4 Models

4.1 Baseline

4.1.1 Random Classification

As the most fundamental baseline, we consider the case of random classification. Given that our dataset is balanced, this model randomly assigns a class to a test sample, with equal probability. Therefore, there is one in ten chance that the prediction made by this model is correct. This model was employed as a basic sanity check for other baseline models. Thus, if any of the other models perform worst than random classification, then they have probably been implemented incorrectly.

4.1.2 Gaussian Mixture Models

The Gaussian Mixture Model (GMM) was employed as a baseline since it is a standard pattern recognition approach and is widely used in literature for modeling audio features for classification tasks. We assume that each of the ten music genre correspond to a GMM model parameterized by the number of mixture components K, the component weights $\phi_{i=1\ldots K}$ and the component mean $\mu_{i=1\ldots K}$ and covariance $\Sigma_{i=1\ldots K}$. We run the following experiments with GMMs to construct a baseline and to understand the impact of different features on the classification task.

1. **GMMs with all the features:**
   In these set of experiments, we use all of the audio features per segment to model the Gaussian mixtures. Therefore, the number of features input into the model is 25, that is, 12 timbral features, 12 chroma features and one loudness feature. However, the challenge in this case was to combine information from different segments of a single track into one vector. To this end, we attempted two approaches: (i) compute the mean feature vector over all of the segments for a single audio file or (ii) treat each segment as an independent sample and input a collection of segments into the Gaussian mixture model instead of a collection audio tracks.

   The former approach is somewhat weaker because by averaging over features, we are losing the temporal relationship between and within segments, which might be crucial to

the performance of the GMM classifier. The latter approach also loses some information by treating each segment as independent. However, since it does not discard raw data from the segments, it is potentially more powerful. The downside of this approach is that it is computationally heavy and takes longer to train. We implement Gaussian mixture components with full covariance, where the parameters are initialized using the K-means algorithm with multiple initial points.

2. **GMMs with timbral features:**
   In order to understand how individual set of features affect the model, we train a GMM that is modeled solely on the timbral features. In this experiment, we treat each segment as an independent sample and train the Gaussian mixture with a collection of segments. Since there are a total of 120 segments per audio track and there are 40,000 audio tracks, the total number of training samples for this experiment would have been 4,800,000. Since 4,800,000 samples take a really long time to train, we considered only 20 segments per track and thus, input 800,000 segments to train the GMM. To compute the labels of the test/validation audio file, we first compute the log probability of each segment under the GMM used to model class g, according to Equation 1. We then, average the log probability for all of the segments and compute a score representing the probability of the entire audio track belonging to class g, as shown in Equation 2. The class label of the audio track corresponds to the class with the maximum score s_g for that track.

$$p_g(x) = \sum_i \lambda_i p(x|\theta_{i,g}) \tag{1}$$

$$s_g(x) = \frac{1}{T} \sum_{t=1}^{T} log p_g(x)) \tag{2}$$

3. **GMMs with pitch features:**
   In order to understand how pitch content affects music genre classification, we construct a GMM model based only on the pitch features. The approach employed in this case is identical to the approach used to model GMMs with only timbral features. Like the timbral features, the labels for this model are computed from Equations (1-2) as described in the previous section.

4.1.3 SVM

As a baseline, we wanted to use a standard non-neural network based classifier and so trained an SVM using the complete feature set. The input vector was 2D: Data x 3000 features. We built the standard SVM using sklearn library and did not use any non-linear kernels.

4.2 Neural Network

We approached genre classification similar to an image classifier because the spectrogram and chroma features for 120 segments represents the audio content pictorially. So, the first approach for the neural network model was a Convolutional Neural Network which is commonly used in image classification and has also been used for spectral based audio classification. Along with learning the spectral image, we were also interested in learning how the model changed over time, i.e. within the 120 segments which requires temporal data. This motivated us to try a Convolutional recurrent neural network. All the models were built using Keras and trained using Google Collaboratory's K80 GPU.

4.2.1 CNN

We used CNN with three 2D convolutional layers with kernel size of (2,2) and 2D Max pooling with pool size of (4,4) summarized in Fig.1. We used all the 25 features and 120 segments per music data for training, making the input size of Batch x 120 x 25 x 1. ReLu activation was used in all the hidden layers and softmax activation was used for the output layer to predict probabilities per class. We validated on 10% of the training set and used Adam optimizer with default settings and cross entropy loss function. The model had a total of 774,410 trainable parameters and was implemented using the

CNN	Activation
Input Layer	
Conv2D	ReLu
Conv2D + Max Pooling 2D + Batch Normalization + Drop out	ReLu
Conv2D + Max Pooling 2D + Batch Normalization + Drop out	ReLu
Flatten	
Dense Layer	ReLu
Dense Layer	ReLu
Dropout	ReLu
Output Layer	Softmax

Figure 1: CNN layers

sequential function in Keras. We used hyper parameters like batch normalization and dropout layers to avoid over fitting. The model was trained for 20 epochs and took fifteen minutes to train.

4.2.2 CRNN

The CNN from the above section was modified by adding one more layer of 2D convolution layer and two layers of RNN cell. RNN cells help in accounting for relative change in the features. For the RNN cell, we used 2 layers of Long short term memory cell (LSTM). The activation used is tanh and hard sigmoid for recurrent step with 'he_normal' kernel. The model is summarized in Figure 2. We used Relu activation for the CNN layer and softmax for the output layer. To feed the output of CNN to LSTM, we had reshaped the vector to be 2D instead of 3D for CNN. The model was implemented using functional API providing the flexibility to input CNN tensors to LSTM unit cells. To avoid over-fitting, we employed an identical approach to the one described in the above section and trained the model for 25 minutes. As in the case of GMMs, we analyzed the model using all the features and then separately for timbral features and chroma features. We looked at the accuracy and confusion matrix to see how the different features affected the neural net.

CRNN	Activation
Input Layer	
Conv2D	ReLu
Conv2D + Max Pooling 2D + Batch Normalization + Drop out	ReLu
Conv2D + Max Pooling 2D + Batch Normalization + Drop out	ReLu
Conv2D + Max Pooling 2D + Batch Normalization + Drop out	ReLu
Reshape	
LSTM layer	tanh (forward) and hard sigmoid (Recurrent)
LSTM layer	tanh (forward) and hard sigmoid (Recurrent)
Dense Layer	ReLu
Dense Layer	ReLu
Dropout	ReLu
Flatten	
Output Layer	softmax

Figure 2: CRNN layers

5 Results

Table 1 shows the evaluation metrics of different classifiers on the modified Million Song Dataset used for this project. Each file is represented by a time series of feature vectors, one for each analysis window. Segments from the same audio file are never split between training and validation data in order to avoid false higher accuracy due to the similarity of feature vectors from the same file. For all the classifiers, we compare their accuracy, training times, F1 score and analyze the confusion matrix. Overall, CRNN gave the best performance.

Table 1: Evaluation Metrics for different models

Model	Accuracy	F1 Average	Training Time(mins)
CRNN training	63.85%		25
CRNN testing	64.47%	33.75	
CNN training	58.6%		15
CNN testing	56.1%	29.4	
GMM Training	51.20%		8
GMM Testing	51.05%	26.76	
SVM Testing	30%	15.7	70
Random	10%	5.11	1

Table 2: Differrnt GMM models

Metric	All features-averaged	All features	Timbral	Pitch
Accuracy Training	51.20%	41.22%	42.11%	27.46%
Accuracy Testing	51.05%	40.67%	41.93%	27.03%
Mean F1 score	26.76	20.97	21.87	3.81
Training Time (mins)	8.33	53.17	44.55	41.12

5.1 Random Classification

Since random classification involves randomly assigning a class to the test samples, the classification accuracy was an average of 10%. This result is reasonable, because there is one in ten chance that a specific audio track has been classified correctly.

5.2 Gaussian Mixture Models

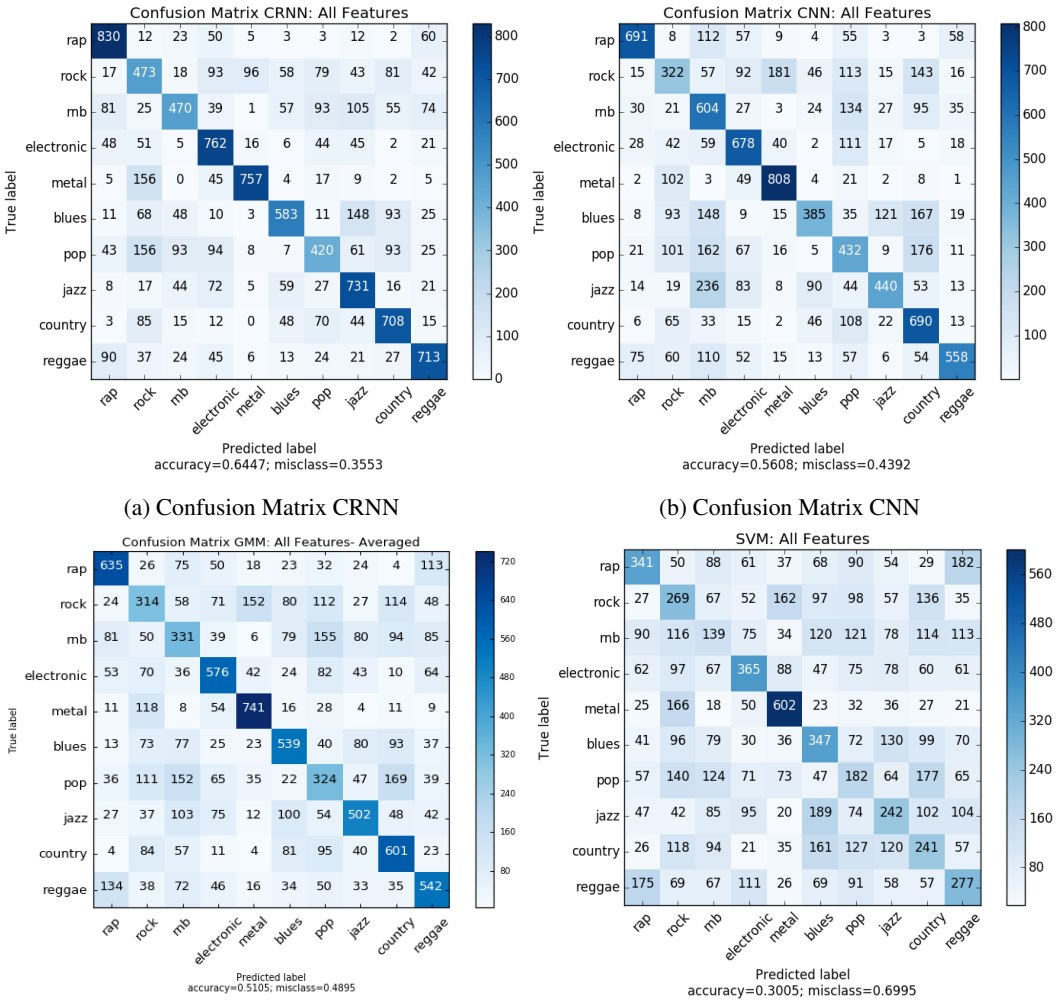
Table 2 summarize the accuracy, training times and F1 scores for the three GMM models. In all three cases, the model was trained using five fold cross validation, where the dataset was randomly partitioned such that 10% was used for validation and 90% was used for training. The reported accuracy metrics are the average accuracy over all five iterations. For models where we treat the segments independently (instead of averaging over them), we only consider 20 segments per audio track and not all 120. This is done to reduce training and evaluation time. The classifier may perform better if more segments were added. After tuning the hyper-parameters by cross validation, it was observed that all the models perform best when the number of mixture components is 5 and the components have a full covariance structure. To summarize the results, the GMM model with all the features averaged over all segments of an audio track performs the best, followed by GMM models for all features over independent segments and timbre features. It is also worth noting that the best performing model takes significantly less amount of time (about $1/6^{th}$ of the time) to train. The classifier based only on pitch content performs the worst. The confusion matrices for the best performing GMM model is represented in Fig 3 and the confusion matrix for the other experiments can be found in the Appendix.

5.3 Support Vector Machine

The accuracy of SVM was lower than GMM but better than the random classification. We did not optimize SVM, but by analyzing the confusion matrix we observed that it mainly worked for metal music and was confused in most of the other genres.

5.4 Neural Network

The CRNN architecture performed better than the CNN model though it took longer to train. Both the networks showed similar trends in terms of precision and recall. Confusion matrix for CNN is provided in the Appendix. Though the recurrent layers benefited Rap class the most with a positive impact on recall values. All the models including baseline performed best on metal, but CRNN



(a) Confusion Matrix CRNN

(b) Confusion Matrix CNN

(c) Confusion Matrix GMM

(d) Confusion Matrix SVM

Figure 3: Confusion matrix of top performing models and baseline

had the best performance for Rap boosted by the increase in the recall value. Rock was the most misclassified genre.

We also looked at impact of different features on the CRNN by training models using Timber and Chroma features separately. The result is summarized in table 4. We found that CRNN with timber features had comparable accuracy to CNN models using all the features. Though CRNN based only on chroma features performed similar to GMMs. Since, the loudness function did not provide a 2D image representation of the audio, we tested the feature using a 3 layer neural net and found the performance to be very low ($< 10\%$)

6 Discussion

From all of the results above we can observe that the models with just timbre features perform relatively well when compared to models using all the features. In contrast, the models with just pitch content perform poorly. Therefore, it seems like timbre features alone contain a lot of information which might be sufficient to categorize music into different genres. Timbre features work well when the music contains a lot of different and distinct instruments. Metal music seems to be one of the most accurately classified genres across all models though CRNN gives best performance for Rap. This maybe because the distinct vocals in metal music makes the timbre of that genre stand out as

Table 3: CRNN performance per class

Class	Precision	Recall	F1
Rap	73.06	83	77.72
Rock	43.8	47.3	45.48
RnB	63.51	47	54.02
Electronic	62.36	76.2	68.59
Metal	84.39	75.7	79.8
Blues	69.57	58.3	63.44
Pop	53.3	42	46.98
Jazz	59.97	73.1	65.89
Country	65.62	70.8	68.11
Reggae	71.23	71.3	71.26

Table 4: Different CRNN models

Metric	All features	Timbral	Pitch
Accuracy Training	63.85%	57.3%	49.3%
Accuracy Testing	64.47%	57.1%	50%
Mean F1 score	33.75	29.9	26
Training Time (mins)	25	15	15

compared to others. Another major observation is that pop, rock and RnB are the most mis-classified genres and are often classified as each other or metal. One explanation for this observation is that pop, rock and RnB often sound similar and might be characterized more by cultural factors and historical periods than by acoustic features. In this case, having textual data such as year of release and artist might help improve classification accuracy.

An interesting trend to note in the GMM implementation is that the model that averages over all the segments with all features does better than the model that treats each segment independently for all features. Intuitively, the latter should do better because it contains more raw information and time series data for the model to work with. However, the problem arises when the pitch in a segment is zero. Thus, the variance of the Gaussian accounting for the pitch is close to zero, resulting in unstable behavior of the model, which detriments performance. When we take the mean over all of the segments, the pitch content is less likely to be zero, resulting in a more stable and slightly better performing model.

When implementing the GMMs, one of the challenges was to combine temporal data across all the segments of an audio track. In order to capture this temporal relationship, we attempted to use techniques such as Dynamic Time Warping (DTW) to measure similarity between two segments of an audio track. The similarity metrics between segments were then used as features for the GMM instead of raw MFCC and pitch content. However, this technique ended up being computationally inefficient and did not increase classification performance significantly. Another technique that seemed somewhat common was to use K-means to cluster the segments and use the centroids of these clusters as features for the Gaussian Mixture model. The rationale behind this approach seemed like it attempted at a more "intelligent" averaging of the segments rather than just taking the mean over all segments. Like the DTW technique, this approach also ended up being computationally heavy and did not yield results due large training times.

The performance of all our models is limited by the fact that we did not have access to raw audio and are thus restricted to only the extracted feature. Therefore, we had no control over when to segment our audio tracks to optimally extract the different features. Conventionally, audio segmentation for genre classification is dictated by the number and occurrence of beats in time. The number of beats per minute further depend on the genre of music. Thus, ideally, our features should vary in the number of segments per track and the length. However, this information is lost when the data is restricted to a fixed number of segments that are similar in length.

7 Conclusion

Music genre classification is a challenging problem, mainly because the distinction between genres is vague and often subjective. For this classification task we used a balanced subset of the Million Song Dataset which consisted of the timbre, pitch/chroma and loudness features for each of the 120 segments of a song. As part of our approach, we established a set of baseline models. The first of these models included random classification, which was implemented to measure the worst case performance, which evaluated to around 10% accuracy. Our most prominent baseline was the Gaussian Mixture Model, where each genre corresponded to a GMM. The probability of an audio track under each model was computed and the label for the track was generated by picking the class with the highest probability. Our best GMM classifier was trained over all features averaged over all segments and garnered an accuracy of 51.05%. We also trained a Support Vector classifier as a baseline. However, this model did not fare too well, providing an accuracy of only 26.64%. We then implemented CNN and CRNN neural network models specifically designed for the genre classification task. Both models outperformed all of the baselines and CRNN performed the best with an accuracy of 64.47%. It was also observed that performance with just timbre features came close to when all the features are used for the CRNN and GMM models. In contrast, pitch features perform poorly as compared to models with just timbre data and all features. Therefore, it can be inferred that timbre data is more important for music genre classification than pitch.

In conclusion, we developed and implemented music genre classifiers that surpassed the performance of all of our baselines and obtained comparable performance to other models in literature used for the same task. We also concluded that timbral features provide more information about music genre than chroma or pitch features. As part of future work, we can look at obtaining raw audio data and extracting our own features to characterize different genres more accurately. Furthermore, incorporating textual data such as year of release, artist and lyrics might also help improve performance, as they might help characterize historical, sentimental and cultural factors associated with different genres. Another line of work can look at using unsupervised learning approaches, to have the system autonomously learn subtle differences between genres, which might not be easily detected by supervised learning approaches.

References

Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. 2017. Convolutional recurrent neural networks for music classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2392–2396.

Sander Dieleman, Philémon Brakel, and Benjamin Schrauwen. 2011. Audio-based music classification with a pretrained convolutional network. In *12th International Society for Music Information Retrieval Conference (ISMIR-2011)*. University of Miami, 669–674.

T Jehan and D DesRoches. 2011. The echo nest analyzer documentation. *Prepared by: September* 2 (2011).

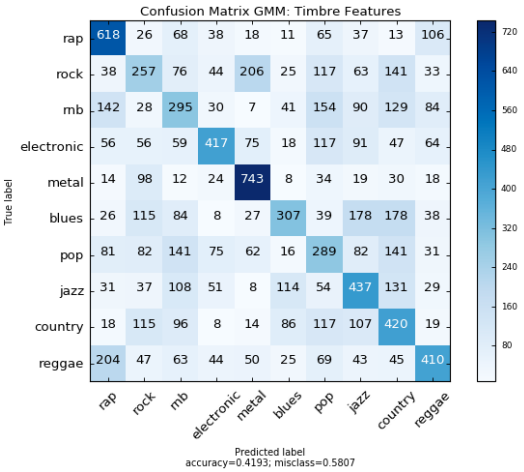
Alexander Schindler, Thomas Lidy, and Andreas Rauber. 2016. Comparing shallow versus deep neural network architectures for automatic music genre classification. In *9th Forum Media Technology (FMT2016)*, Vol. 1734. 17–21.

Hendrik Schreiber. 2015. Improving Genre Annotations for the Million Song Dataset.. In *ISMIR*. 241–247.

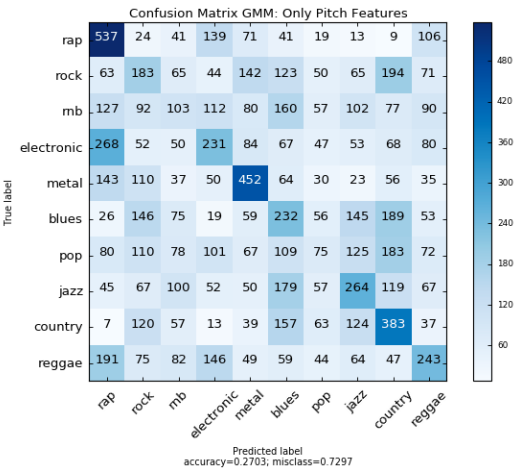
Bob L Sturm. 2012. A survey of evaluation in music genre recognition. In *International Workshop on Adaptive Multimedia Retrieval*. Springer, 29–66.

George Tzanetakis and Perry Cook. 2002. Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing* 10, 5 (2002), 293–302.

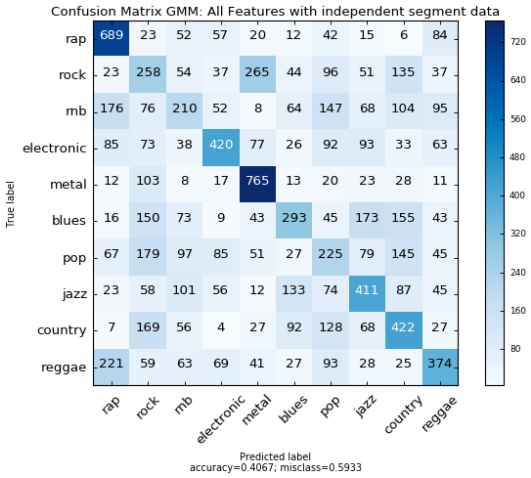
8 Appendix



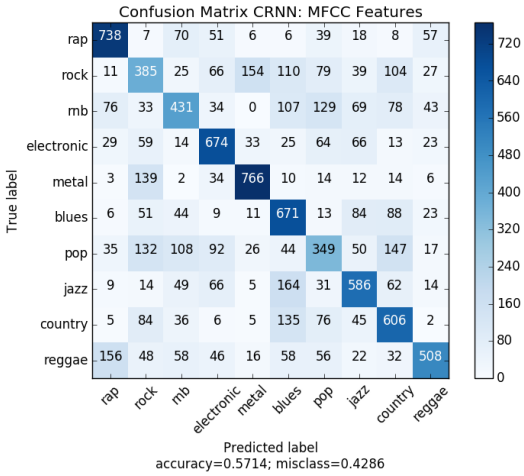
(a) Confusion Matrix for GMM trained using only timbre features



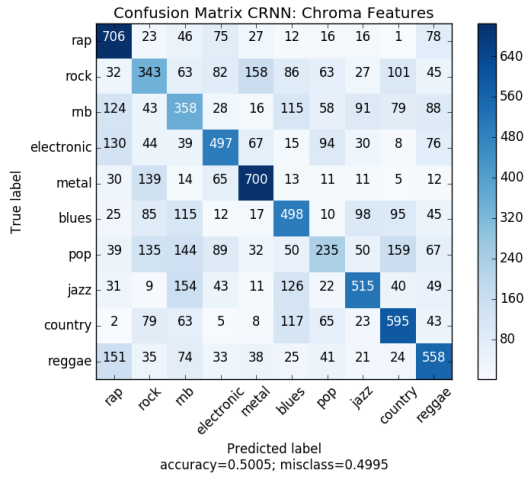
(b) Confusion Matrix for GMM trained using only pitch features



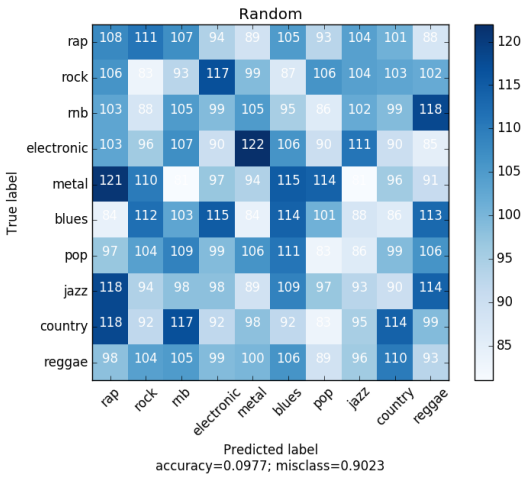
(c) Confusion Matrix for GMM trained using all features where each segment of an audio file is treated as an independent data point.



(d) Confusion Matrix for CRNN trained using only timbre features



(e) Confusion Matrix for CRNN trained using only chroma/pitch features



(f) Confusion Matrix for Random Classification