Music Genre Detection

Rahul Rammohan Mandical
UBID - 50363222
rrammoha@buffalo.edu

Arnav Sinha
UBID - 50419372
arnavsin@buffalo.edu

Abstract

With the advent of the internet, we saw a huge movement of music from being stored on records and tapes to the cloud. Currently, everyone gets their music from mega websites such as Spotify or SoundCloud. These corporations handle millions of songs with around a hundred new ones being uploaded each week. There is a need to tag genres to these songs efficiently and accurately without human intervention. This comes in the form of Deep Learning models. The paper attempts to produce models which can identify the genre by using both audio and lyrical features. Deep Learning algorithms such as Deep neural networks, Convolutional neural networks, and Recurrent neural networks have been used to their full effect on the Free Music Archive Dataset. In the process, we will also measure the impact of lyrics on the model as well as determine the better approach with respect to vector size and song duration.

1 Introduction

Merriam-webster dictionary defines genre as a category of artistic, musical, or literary composition characterized by a particular style, form, or content. In our case, we will look at genres of music. Rock, Hip-hop, Electronic, etc. are some of the more popular genres. Any song that exists in this world contains a primary genre and a secondary genre. We will be focusing our efforts on identifying the primary genre. The main aim and primary objective are to build feasible models which can accurately identify the genre of the music. We will use audio features of the actual song extracted in the form of MFCC features and Lyrical features converted to themes. As part of our secondary objectives, we will experiment with the effect of lyrical data on the Neural Network models. This is tested by comparing models which used both audio and lyrical data versus models which used only audio data. We shall also Identify if it is better to have longer songs but less in number or have shorter songs which are more in number i.e. 250 vector size vs 125 vector size. This is verified by splitting each audio into 5 vs 10 parts and training the networks.

One can ask the question, why even try to identify the genre? There are a lot of benefits to knowing the genre of the music. In academic terms, the genre can not only help create new music but also understand the intrinsic features that help make music special. From a commercial perspective, huge corporations such as Amazon Music or Spotify can use a system to accurately tag and group huge amounts of songs swiftly without the need for human intervention. By using DL models, we can definitely and efficiently tag the songs in an accurate manner.

The dataset that is used in this paper is obtained from FMA or Free music archive (18). We downloaded 25000 songs and then extracted the MFCC features using Librosa. For the lyrics, we used Genius API and converted them into theme features. We ended up with two datasets after merging, one with a 250 vector size and another with a 125 vector size. K-nearest neighbors, DNN, CNN, RNN-LSTM, RNN-GRU, and hybrid models are trained on 250 vector datasets. For the 125 vec dataset, we trained the best models of DNN, CNN, RNN, and hybrid from 250 vector datasets and compare the results. In order to measure the impact of Lyrical features, we trained models on 250 vector datasets without theme features. The entire process can be viewed in this flowchart 1 . The

following section provides insight into other papers that have attempted to solve the same problem with different techniques.

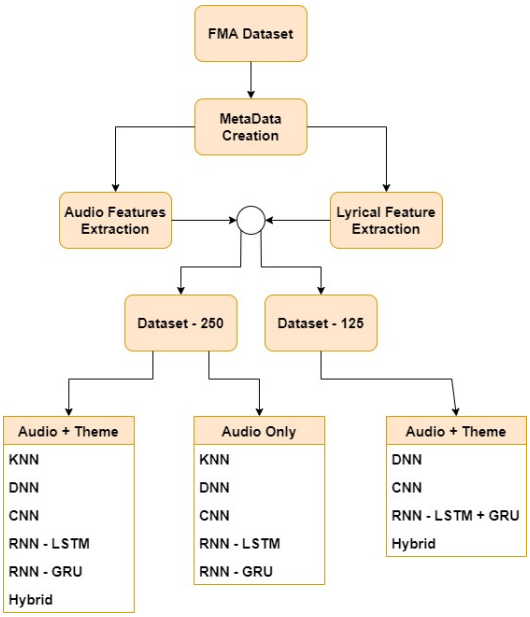

Figure 1: FlowChart

2 Literature Survey

We will first take a look at some of the papers that tried to solve the same problem and then we will deep dive into one of the papers

2.1 Related Work

While this problem is not as prevalent as something like Fake News Detection, there is still an ample amount of research done. We can divide the papers into three categories. Category one papers solve the problem by only looking at audio features. Category two on the other hand only looks at Lyrical Features. Category three papers try to combine multiple features to build the models. Our paper lies in the third.

Audio Centric Papers or Category 1 papers are the most abundant. Most of the papers use a spectrograph or its variants while some of them also use MFCC and its alternatives. With regard to models, multiple types of Deep learning algorithms have been used along with some unique architectures. **Liu *et al.***(1) uses a novel CNN architecture that takes into consideration the long contextual information. The paper uses Mel spectrogram to represent the audio. They experimented on multiple datasets including GTZAN and produced respectable results of 93.9% accuracy.

Like **Liu *et al.***(1), **Dong *et al.***(3), and **Zhang *et al.***(17) use CNN as their main algorithm. **Dong *et al.***(3) uses an ensemble approach where the genre is predicted by splitting the audio and combining the results of the model for all the parts. They also claim to outperform human predictions. **Zhang *et al.***(17) on the other hand takes a more common approach and produces two models one using pooling and the other using shortcut connections similar to residual learning. **Zhang *et al.***(17) also trains its models on the GTZAN dataset.

Feng *et al.*(2) uses an STFT spectrogram to represent the audio and creates a model that is a combination of CNN and RNN (B-GRU). They produce similar results to our models but on GTZAN dataset. While all these papers tries to classify genre, **Hsu *et al.***(4) takes a different approach and tries to classify the subgenres of EDM music. They use Mel-Spectrogram and Tempogram Features. They also use a unique short chunk CNN + ResNet models. **Heo *et al.***(5) uses frequency sub bands aggregation method which divides the input spectrogram along frequency bandwidths. Their

framework is based on ECAPA - TDNN, where all the layers that extract short-term features are affected by the layers that extract long-term features. They train their models on much larger dataset, but the results leave a lot to be desired for.

Most papers mainly stick with balanced Datasets, but **Liu,Xiaokai *et al.***(9) proposed a novel mechanism named Multi-instance Attention (MATT) which is used to boost performance for tail classes. This helps target imbalanced datasets where certain genres do not have enough audio files. **Chathuranga *et al.***(13) on the other hand doubles down on using ML algorithms for their models. They work on GTZAN and ISMIR2004 by using SVM Classifier with polynomial kernel functions. The weak learner is compensated by using Adaboost and Ensemble classifiers. The models have provided a surprising accuracy of 78% and 81% for the datasets used, Finally **KarunaKaran *et al.***(16) uses a two-phase hybrid classifier to overcome the blurry classification of the genres. Similar to our paper they have worked on GTZAN and FMA dataset. Our models have outperformed their models for FMA dataset.

Using Lyrical Features only is a niche category in our problem. **Tsaptsinos *et al.***(6) uses a Hierarchical Attention Network on the lyrics to exploit the layer structures, we will deep dive into this paper in the later section. **Araujo Lima *et al.***(10) used BLSTM along with SVM and Random forest combined with multiple word embeddings techniques. However, these models were built to identify Brazilian lyrics specifically

Instead of concentrating on only one aspect, these papers have created models to encompass both audio and lyric features. **Oramas *et al.***(7) takes a distinctive approach where they use cover image, text reviews and audio tracks. They also present their MuMu dataset which contains over 31k albums with 250 genres. The paper provides a different perspective to classification by using different Data modalities. **McKay *et al.***(11) takes a step further by considering cultural features such as web searches along with the usual features. They also showed how features extracted from lyrics were less effective compared to others.

Finally, **Mayer *et al.***(12) which tends to lean more heavily towards lyrics. They use advanced lyrics features such as the frequency of certain rhyme patterns, several parts-of-speech features, and statistic features such as words per minute (WPM) along with regular audio features. This paper sticks to ML models using KNN, Naïve Bayes and SVM. The results range from 40 to 60% for various datasets which clearly shows the short comings of ML models

While the previously mentioned papers have used a plethora of DL algorithms, this paper confines itself to three most important algorithms DNN,CNN and RNN. In the next section, we will provide an in-depth look into one of the papers.

2.2 Deep Dive into "Lyrics-Based Music Genre Classification Using a Hierarchical Attention Network"

This paper in particular caught our attention among all the other papers. The techniques used in this paper is very well thought out and can complement our paper as well. It takes a complete NLP approach to solve the issue of genre detection. The paper uses only Lyrical data similar to how we have used except the preparation and pre-processing are quite different. Some of the biggest contribution from this paper is that HAL or Hierarchical Attention Networks can be used to take advantage of the hierarchical structure of the sentences and we can clearly visualize which words are most important. We believe that knowing what makes are most important to define genre greatly benefits the models.

The paper uses a very huge Lyrical Dataset acquired from LyricFind which contains around 1,039,151 songs. While there were no genres, the author was able to acquire them with the help of iTunes. This however severely reduced the number of songs due to lack of missing data. Instead of simply creating their own dictionary, the author has re trained their words over the GloVe embeddings which they claim helps boost the metrics. They create two datasets one with 20 genres and other with 117 genres.

The main algorithm used is Hierarchical Attention Networks. It is based on bidirectional gated recurrent unit with attention applied to its output. We humans understand the meaning of sentence by reading the words and understand the paragraph by understanding the group of sentences. Similarly, the network tries to understand the context of the paragraph by getting values for sentences and sentences from values of words. This is especially helpful for Homonyms.The BRNN part learns

the meaning behind those sequences of words and returns a vector. The attention network gets the weights for each vector using its own shallow neural network. Then it aggregates the representation of those words to form a sentence vector. This weighted sum embodies the whole sentence. Similarly, we do the same except for the sequences of sentences to understand the paragraph.

The paper compares HAL with other algorithms Logistic Regression and LSTM. While the overall performance of all models are in the range of 40 –50%, we see that HAL as outperformed the other algorithms in both 20-genre dataset and 117-genre dataset.The metrics show that while the models are good, perhaps lyrics are just not enough for classification of genres. The best way to improve the models is to introduce extra features in the form of audio representation of external cultural features.

According to us, The most important part of this paper is the Attention visualization. By observing the weights for the words, we can extract what kind of words have the most impact to the genre.The author mentions that words such as 'cause' and 'gonna' were mostly weighted in hip hop songs and 'your' and 'you' in rock songs along with 'woh' and 'ooh'. This feels in line with the current trend in music. However, the model is not perfect as it has weighed heavily on empty space. This could either be the importance of silence and an issue in dataset.

The main takeaway from this paper is that this paper solves one of our sub-problems about how to extract the most important part of Lyrics. We use a theme approach, but this HAL approach makes sense too. Perhaps in the future, we can substitute our theme part with a HAL model to give us the most important words from each song. This can be extracted from the weights of the attention network.The idea of retraining Glove word embeddings is also something we could have experimented on in our project.

3 Dataset Preprocessing

Before choosing FMA – medium dataset, we explored other options. The most commonly used Dataset was the GTZAN dataset. It contained 1000 songs distributed among 10 genres. The biggest issue with this dataset is the lack of data regarding the track and artist and, couple that will less number of songs, we rejected it. We also tried building our own dataset by using Spotipy API, unfortunately we were not able to acquire all the songs due to legal and copyright issues. The songs we did acquire were not perfect. The FMA – Free music archive provided us with three options, a small dataset, medium and large. The dataset was ripe with meta data information and the medium dataset had 25000 songs which was ample for us. So we chose FMA – medium as our dataset and downloaded the songs.

3.1 MetaData Creation

We first built our own metadata table which contained track, artist and genre information. This was achieved by using the files and songs provided by FMA. We will refer to this table for all our meta information throughout the project. Since this paper uses both Lyrical and Audio Data, we needed a way to download the lyrical features and extract the audio features.

3.2 Audio Features

With regard to the audio features, we will be using MFCC or mel-frequency cepstrum coefficients. To generate MFCC, we first create a spectgram called STFT or short time Fourier Transformation. On this , we apply mel scaling and Discrete fourier transformation to get MFCCs. We will be using Librosa library which can automatically generate for us the MFCC values for the track.

We shall use the first 15 coefficients; anything present in higher frequencies will have more noise than useful information. For each song, we shall divide the song into more parts. This is done by keeping track of samples per segment. Since each song is of 30 seconds long. We need to decide how many parts as it will have an impact the vector size. When we split the song into 5 segments each of 6 seconds. We will be left with MFCC feature of shape (250, 15), here 15 refers to the 15 MFCC coefficient while 250 refers to the number of frames. If we split the song into 10 segments each of 3 seconds then we get shape (125, 15). Less number of frames is compensated with a greater number of song parts. We generate both 250 vec and 125 vec features.

Table 1: Dataset Specifications

	250 Vec	125 Vec
Rock	4775	9550
Hip-Hop	4130	8260
Experimental	4035	8070
Folk	4030	8060
Electronic	3990	7980
Pop	3035	6070
Instrumental	2905	5810

3.3 Lyrical Features

Since we now have audio features, we need to download the lyrical features. This is accomplished using Genius API. The api was not able to find lyrics for all songs, so some of them had to be dropped. We then use Fasttext(19) (20) language identification model to identify the language and drop all non-English songs. Post that we do some cleaning and tokenizing. Now using the whole lyrics would not be good as it contains a lot of extra data. We wanted to extract the most important part of lyrics that is the theme, this comes in the form of certain words that have the biggest impact. We created A CBOW(21) dictionary that is trained on all lyrical data to get the word embeddings. We removed stop words (words like 'a, 'the' etc) using NLTK's stopwords module from each lyric. From the resultant list of tokens, we took the 5 most repeating tokens. We used the CBOW model's word embeddings for these 5 tokens as our theme features. This was done for both 250 vec dataset and 125 vec dataset.

3.4 Final Dataset Creation

We finally merge both audio and theme features to form both of our datasets. At this point we shall assume that it is better to have 250 vector size since the song duration is longer thus allowing for better representation. We shall use 250 vec dataset as our main dataset and build models on it while 125 vec datasets will be used as secondary datasets. For training models on audio only, we will drop the extra theme features.In total there is 26900 songs in 250 vec dataset and 53800 songs in 125 vec dataset. The genres present include Rock,Hip-hop,Experimental, Folk, Pop and Instrumental. The specifications of the dataset can be found in Table 1.

4 Models and Experiments

While there are many ways one can build models, we decided to go the route of tried and tested methods. We chose KNN to act as our baseline ML model. DNN and CNN will also be used along with RNN. For RNN, we trained models on both LSTM and GRU, this allowed us to do a minor comparison between the algorithms. We also experiment with hybrid algorithms which is a mixture of CNN and RNN. Before arriving on split neural network architecture, we experimented with single input Neural networks where the concatenated data of audio and theme were used together, this failed due to the model thinking they both are the same thus making it difficult to learn. The split network creates an isolation between the features for better training. We also found out that Logistic regression had a hard time converging with our dataset.

For each model, we split the dataset into train test and valid with ratio 80:10:10, we would train and test the models on train and test dataset and final evaluate on valid dataset to simulate real life. We also one hot encoded our output.

4.1 K Nearest Neighbor (KNN) Classification

KNN here serves as a ML baseline model, while we cannot expect KNN to represent the whole of ML family of algorithms, it does serve as a comparision for us. One of the advantages of KNN is that it can work well with both non-linear data and multi-class classification.

KNN performs classification by plotting all training samples in an n-dimensional space (n being the number of features). To predict the class of new data, it plots the new data in the same space. It then computes the distance from each training sample and determines the k nearest training samples with

value of k based on user input. The class of the new data is predicted to be the same as the most popular class in the k neighbors found. The algorithm performed poorly in the music genre detection problem space due to the high number of dimensions of the training samples.

4.2 Deep Neural Network (DNN)

Deep Neural Networks (DNNs) is one of the most simplest neural networks used in Deep Learning. Its working takes inspiration from the human brain and similar to that, the foundational unit is a neuron. These neurons are interconnected and pass information between themselves to learn to generate an output for a given input.

A basic architecture of a DNN comprises of an input layer, an output layer and one or more hidden layers in between. The input layer neurons take their values directly from the training sample features. The hidden layers also called as dense layers comprises of specific number of neurons each of which are completely connected to all the neurons in the previous and next layer. The output layer has neurons for each output class and connected to all the neurons in the last dense layer. The learning in DNNs happens in two alternating steps – Feed Forward and Backpropagation performed iteratively until loss is minimized. In each iteration, the neurons apply a set of weights and biases to the input and before sending to the next layer, adds an activation function. A loss function is used to calculate how far away are the predictions of the network with respect to true values. The main idea behind DNN is minimize this loss functions. The learning process takes place in back propagation where the weights and biases are updated based on the loss and gradient. The continuous updating of weights and bias until the loss is minimized is nothing but training. The idea of trying to find the global minima by calculating the loss is called gradient descent. We have used batch gradient descent for all our models. The optimizer used is Adam optimizer which is a gradient-based optimizations, based on adaptive estimates of lower-order moments. The main loss function used is categorical cross entropy since we have one hot encoded our output.

For the 250 vector dataset, the best performance for audio + theme features models was given by the DNN model. It was split neural network with a split each for audio features and theme features and early flattening. The audio split compared to the theme split had more dense layer processing but no regularization. The theme split had light dropout regularization. The two splits were then concatenated before being processed through more dense layers and finally feeding into the output layer. The audio data only DNN model performed poorly. The trend of stunted test accuracy is found in the graph. Like the audio + theme features model, it had early flattening followed by 9 dense layers before feeding into the output layer.

4.3 Convolutional neural networks (CNN)

CNN is a specialized neural network that is based on the convolutional operator. The main intuition behind CNN is the use translation in-variance and locality. Translation in-variance refers to when the behavior of the network does not depend on the location of the patch and locality here means that far away neurons have less affect compared to closer neurons. This helps especially with image classification as the surroundings of the subject are more important than the corners of the image. It can also be used for image segmentation and signal processing successfully.

One can use a filter and apply convolutional operator to transform the image, The effect of filter movement can be controlled using stride. Adding padding to the image allows extra space for the filter. As before, we will use Keras and Tensorflow to implement CNN.

Some of the more important layers are Conv2d which handles the actual convolution. Max pooling layers are used to pool i.e down sample the input by taking the maximum value in the window. We also used Batch normalization layer to speed up the process and normalize the inputs. Similar regularizations such as dropout and activity has also been used.

The best model for audio+theme dataset contains three sets of Conv2d,pooling and batch normalization followed by flattening. For theme we have just 1 conv2D and maxpooling. We have interleaved dropout layers too. Post concatenation we have a few dense layers and final output.

The best model for audio only dataset contains 3 sets of conv2d and maxpooling followed by batch normalization and dropout, we then flatten and add some dense layers. The results are similarly insufficient like the other algorithms. Observing the graph, see that it has a natural progression of

loss and accuracy without any discrepancies. But the audio only graph follows the trend of stagnating validation accuracy.

4.4 Recurrent neural networks (RNN)

Recurrent neural network is another special type of NN. One of the reasons why RNN works well with sequential data is that output of the previous computation is used along with the input of the current computation. This means that the model results depend on both input of current epoch and output of previous epoch. This is achieved by RNN having a memory module to process the output. The biggest drawback of this model is that it is very susceptible to vanishing gradients and might have trouble remembering long sequences. To counteract this drawback, we have two approaches, one is the form of Long Short-Term Memory and other is Gated Recurrent Units.

4.4.1 Long short term memory (LSTM)

LSTM address the drawback by using three special gates. The input gate whose main job is to choose what inputs must be saved, Sigmoid function is used to choose the value while tanh function chooses the weightage for it. Forget Gate on the other hand decides what needs to be discarded. It provides a value between 0 to 1. 0 means forgetting and 1 means retaining. Finally the output gate decide what values to let through. These gates with its memory help in remembering the longer sequences however this comes at the cost of training longer and more memory requirement. The main LSTM layer is implemented with keras.

The best model for audio+Theme dataset contained two layers of LSTM on both sides i.e. audio and theme. Post concatenation , we again use two LSTM layers followed by two dense layers. While no regularization was used for this model, the best model for audio contains a dropout layer in the middle. Its starts of with a pairs of LSTM and Dense Layers ending with a triple Dense layer structure. This model specifically took the longest to train, training for 240 epochs. Graph for audio also suffers from the same plateauing issue.

4.4.2 Gated Recurrent Units (GRU)

GRU is another variant of RNN similar to RNN but tries to rely more on its hidden state to transfer memory between the units thus foregoing the need for separate cell state. This comes in the advantages of shorter train times and less memory usage. But it may not be able to remember as long sequence as LSTM.

Here we have only Reset gate and update gate. The update gate as the name mentions updates the hidden state with new information, reset gate on the other hand combines the previous hidden state and current state and chooses what to discard or remember. It can choose to retain partially by providing values between 0 and 1. The hidden state in traditional RNN is replaced by Candidate sate which also has an activation.

While GRU models performed similar to that of LSTM models, we found that best Gru model also contains LSTM layers. For Audio input, we used 2 LSTM layers with a dense layer and for theme input we have two Gru layers with a dense layer. Post concatenation, we have 4 dense layers. We again see a natural rise and fall in the graph. Similar to LSTM, Audio only model contained 2 GRU layers followed by 4 dense layers. The same performance is noticed in the graphs. LSTM and GRU models have both performed similarly with LSTM slightly edging GRU out.

4.5 Hybrid models

Until now we have looked at models which have restricted themselves to a single algorithm. One of the biggest advantage of split neural networks is that it allows to train models with completely different algorithms. By doing so , we hoped to harness the advantages of both sides of algorithms. We attempted this only on audio + theme datasets.

We tried two types of models, one with LSTM for audio and CNN for lyric while the other CNN for audio and GRU for Lyric. The best model is the combination of GRU and CNN. The audio input contains 3 sets of conv2d, pooling and normalization followed by dropout and flattening. The theme input is much smaller with only two GRU layers and 1 dense layer. Post concatenation , we have 4 dense layers.

Table 2: Accuracy for 125 vec Dataset

Models	Train Accuracy (%)	Test Accuracy (%)	Valid Accuracy (%)
DNN	94.83	90.63	90.69
CNN	90.12	86.62	87.57
RNN (LSTM + GRU)	95.99	90.84	91.34
Hybrid (RNN + CNN)	94.18	89.87	90.11

One would expect LSTM and CNN combo to do better since LSTM can remember more and previous results support it. But GRU and CNN was able to learn better only my small margin.

4.6 Models for 125 vec dataset

For this dataset, we specially chose the best models from the 250 vec dataset and trained them. As expected, the best model turned out to be an RNN architecture with a combination of both LSTM and GRU. Audio input contained LSTM layers while theme contained GRU layers. Post concatenation was all dense layers.

The worst performance was by CNN model however it was not too far behind. The best DNN and hybrid models showed similar performance. Training these models helps is evaluate on our second objective as well as better understand the impact of vector size for models. The results from these models are much better than expected and with more resources, we can surely push for higher numbers. We shall analyze the results for the models in the next section.

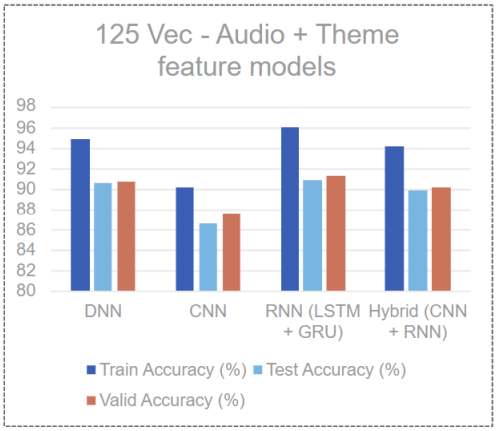

Figure 2: 125 Vec Dataset Accuracies

5 Best models for Audio + Theme

The best models for 250 vec dataset is based on DNN with 85.8% while for 125 vec it was from RNN with 91.34 %. Figure 4 refers to the architecture.

6 Results Analysis

Taking a step back and looking at all the models shows us that theme features have certainly helped. All deep learning models have performed well and as expected the ML models failed. Overall the best performance by sheer valid accuracy is given by DNN which is of 86% for 250 vec Dataset - Audio + Theme.

When looking at Audio only models, we see the clear lack of good metrics. This is due to the model not having enough features to efficiently and accurately learn the dataset. Most of the test and valid accuracies lies in and around 45%. The early plateauing of the models shows that it never generalizes well. Only RNN which is the best here, performed slightly better with 62%, this proves that RNN

does well with sequential data. Table 4 refers to the all values for 250 vec Dataset - Audio only while Figure 3 shows the same in a graphical format.

Unlike audio only mode, audio + theme models do not suffer from any issues, for 250 vec datasets, we see a respectable values around 85%.One would expect RNN to perform best since the sequence nature of songs would be perfect fit for this architecture, but It was slightly overshadowed by DNN. Again, the difference is negligible. CNN performed relatively worse compared to DNN and RNN. We had high hopes for hybrid models, perhaps the strengths of RNN and CNN could be combined but unfortunately there was no specific improvement over other pure algorithm models. Table 3 refers to the all values for 250 vec Dataset - Audio + Theme while Figure 3 shows the same in a graphical format.

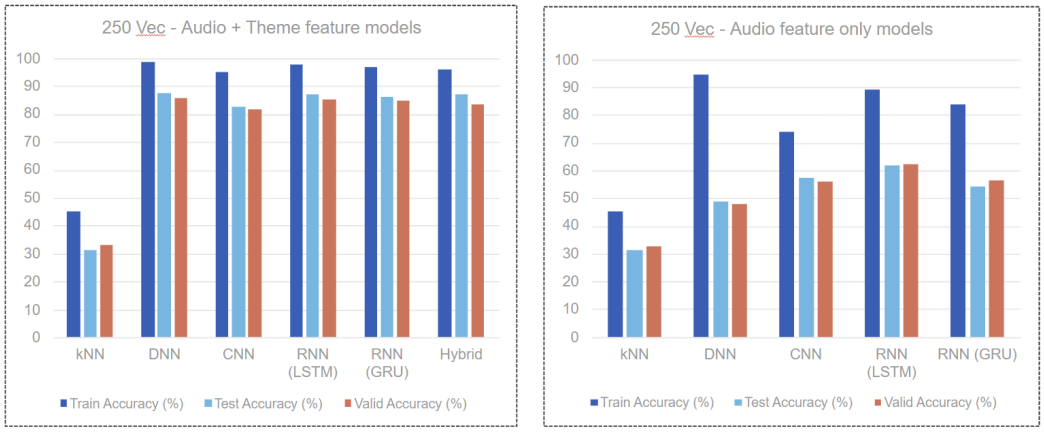

Figure 3: 250 Vec Dataset Accuracies

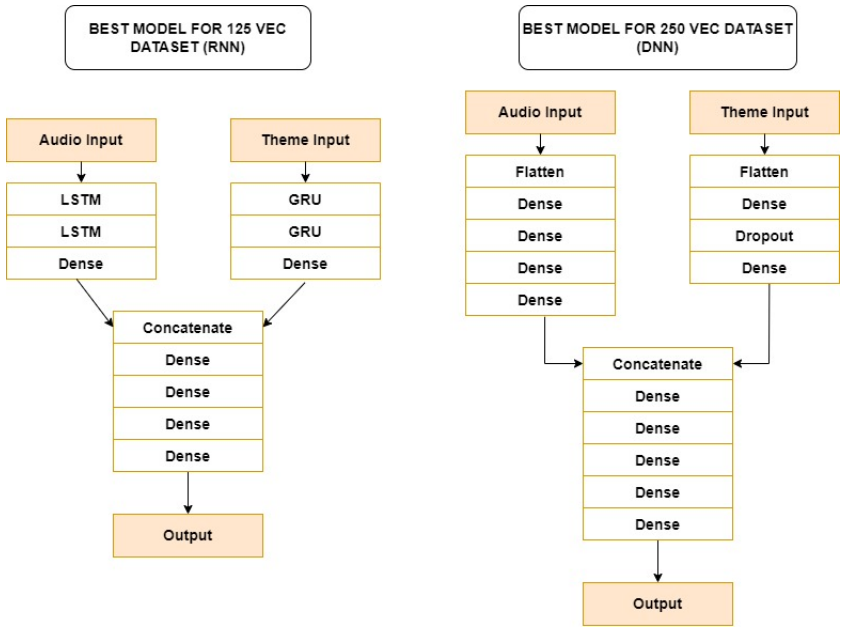

Figure 4: Best Model Architectures - Audio + Theme

As a minor side objective, we compared the performance of LSTM vs GRU, we see that at least for our dataset there is not much difference in metrics. However, we can never say that one is better than the other as there will always scenarios where one outperforms other.

Table 3: Accuracy for 250 vec Dataset - Audio + Theme

Models	Train Accuracy (%)	Test Accuracy (%)	Valid Accuracy (%)
KNN	45.22	31.37	33.01
DNN	98.65	87.55	85.8
CNN	95.2	82.71	81.63
RNN - LSTM	97.84	87.14	85.39
RNN - GRU	96.66	85.95	84.94
Hybrid (RNN + CNN)	95.98	86.88	83.19

Table 4: Accuracy for 250 vec Dataset - Audio only

Models	Train Accuracy (%)	Test Accuracy (%)	Valid Accuracy (%)
KNN	45.12	31.3	32.75
DNN	94.63	48.96	47.69
CNN	73.82	57.21	56.06
RNN - LSTM	89.03	61.97	62.01
RNN - GRU	83.59	54.28	56.36
Hybrid (RNN + CNN)	-	-	-

While 125 vec dataset was considered as our secondary dataset, the models have performed much better than 250 vec datasets. All algorithms see a significant increase in metrics where all except CNN break the 90% barrier. The highest being 91% for RNN model. This model uses both LSTM and GRU for its layers, but DNN and Hybrid models are not far behind at 90%.The results shows us that perhaps it is better to have a greater number of songs than having longer songs with more data. Table 2 refers to the all values for 125 vec Dataset while Figure 2 shows the same in a graphical format.

As for ablation experiments, it comes in the form of adding or removing theme features, it is very clear that these features have a huge impact and is very necessary. Vector size also has a significant impact on the models.

7 Conclusion

This project has shown that deep learning is viable method for building models to identify the genre of the dataset. We have successfully accomplished our primary objective of building viable models. Regarding the secondary objectives, We have shown that it is much better to have Audio + Theme data. It makes sense since more features will help the model most of time except for noisy features. In the beginning, we took a decision to put most of our effort into 250 vec models. We believed that having more information per song would of more help to the models. Unfortunately we were wrong, 125 vec models outperformed 250 vec datasets. This shows that maybe it is better to have more songs.

For the future, we can always expand on 125 vec models for better performance. We can also improve the features by using chords and pitch fluctuations. We could also better use lyrics features instead of eliminating them. Expansion of the project into identifying subgenres is also viable.

All in all, this project has been an eye opener for us. We were able to learn a lot of new concepts and got first hand experience working with them. There is still a lot of potential to expand on this problem and we intend to do so in the future. Access the codes and models here.

References

[1] Liu, Caifeng, Lin Feng, Guochao Liu, Huibing Wang, and Shenglan Liu. 2021. "Bottom-up Broadcast Neural Network for Music Genre Classification." *Multimedia Tools and Applications* 80 (5): 7313–31.

[2] Feng, Lin, Shenlan Liu, and Jianing Yao. 2017. "Music Genre Classification with Paralleling Recurrent Convolutional Neural Network." *arXiv Preprint arXiv:1712.08370*.

[3] Dong, Mingwen. 2018. "Convolutional Neural Network Achieves Human-Level Accuracy in Music Genre Classification." *arXiv Preprint arXiv:1802.09697*.

[4] Hsu, Wei-Han, Bo-Yu Chen, and Yi-Hsuan Yang. 2021. "Deep Learning Based EDM Subgenre Classification Using Mel-Spectrogram and Tempogram Features." *arXiv Preprint arXiv:2110.08862*.

[5] Heo, Jungwoo, Hyun-seo Shin, Ju-ho Kim, Chan-yeong Lim, and Ha-Jin Yu. 2022. "Convolution Channel Separation and Frequency Sub-Bands Aggregation for Music Genre Classification." *arXiv Preprint arXiv:2211.01599*.

[6] Tsaptsinos, Alexandros. 2017. "Lyrics-Based Music Genre Classification Using a Hierarchical Attention Network." *arXiv Preprint arXiv:1707.04678*.

[7] Oramas, Sergio, Oriol Nieto, Francesco Barbieri, and Xavier Serra. 2017. "Multi-Label Music Genre Classification from Audio, Text, and Images Using Deep Features." *arXiv Preprint arXiv:1707.04916*.

[8] Medhat, Fady, David Chesmore, and John Robinson. 2017. "Music Genre Classification Using Masked Conditional Neural Networks." In *International Conference on Neural Information Processing*, 470–81. Springer.

[9] Liu, Xiaokai, Shihui Song, Menghua Zhang, and Yafan Huang. 2022. "MATT. A Multiple-Instance Attention Mechanism for Long-Tail Music Genre Classification." In *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 782–87. IEEE.

[10] Araújo Lima, Raul de, Rômulo César Costa de Sousa, Hélio Lopes, and Simone Diniz Junqueira Barbosa. 2020. "Brazilian Lyrics-Based Music Genre Classification Using a BLSTM Network." In *International Conference on Artificial Intelligence and Soft Computing*, 525–34. Springer.

[11] McKay, Cory, John Ashley Burgoyne, Jason Hockman, Jordan BL Smith, Gabriel Vigliensoni, and Ichiro Fujinaga. 2010. "Evaluating the Genre Classification Performance of Lyrical Features Relative to Audio, Symbolic and Cultural Features." In *ISMIR*, 9:213–18. 13.

[12] Mayer, Rudolf, Robert Neumayer, and Andreas Rauber. 2008. "Combination of Audio and Lyrics Features for Genre Classification in Digital Audio Collections." In *Proceedings of the 16th ACM International Conference on Multimedia*, 159–68.

[13] Chathuranga, Dhanith, and Lakshman Jayaratne. 2013. "Automatic Music Genre Classification of Audio Signals with Machine Learning Approaches." *GSTF Journal on Computing (JoC)* 3 (2): 1–12.

[14] Çoban, Önder, and Işıl Karabey. 2017. "Music Genre Classification with Word and Document Vectors." In *2017 25th Signal Processing and Communications Applications Conference (SIU)*, 1–4. IEEE.

[15] Dai, Jia, Wenju Liu, Chongjia Ni, Like Dong, and Hong Yang. 2015. "'Multilingual' Deep Neural Network for Music Genre Classification." In *Sixteenth Annual Conference of the International Speech Communication Association*.

[16] Karunakaran, Nagamanoj, and Arti Arya. 2018. "A Scalable Hybrid Classifier for Music Genre Classification Using Machine Learning Concepts and Spark." In *2018 International Conference on Intelligent Autonomous Systems (ICoIAS)*, 128–35. IEEE.

[17] Zhang, Weibin, Wenkang Lei, Xiangmin Xu, and Xiaofeng Xing. 2016. "Improved Music Genre Classification with Convolutional Neural Networks." In *Interspeech*, 3304–8.

[18] Defferrard, Michaël, Kirell Benzi, Pierre Vandergheynst, and Xavier Bresson. 2016. "FMA: A Dataset for Music Analysis." *arXiv Preprint arXiv:1612.01840*.

[19] Joulin, Armand, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. "Bag of Tricks for Efficient Text Classification." *arXiv Preprint arXiv:1607.01759*.

[20] Joulin, Armand, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. "Fasttext. Zip: Compressing Text Classification Models." *arXiv Preprint arXiv:1612.03651*.

[21] Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. "Efficient Estimation of Word Representations in Vector Space." *arXiv Preprint arXiv:1301.3781*.