Group 25: Music Genre Classification

Team Name: Niner Miners

Team Members:

Mayuri Kingre

Sayali More

Nishita Kalyanour

Grishma Kalidindi

1. Brief Problem Description

Many companies nowadays use music genre classification for the purposes such as recommendation, or just to recognize the music like shazam. One of the best ways to categorize or classify music is by its genre. For this purpose, supervised machine learning approaches are considered. Music Genre Classification reads the audio files as input dataset and classifies them based on its genre. This can be done using Convolution Neural Networks (CNN). For this purpose, we plan to use Keras which is an open-source neural network library as it supports convolution neural networks and recurrent networks and the combination of two. It runs on top of the TensorFlow, CNTK, or Theano.

In the Music Genre Classification system, the aim of the proposed system is to compare the classic approach of extracting the features using a classifier and using a Convolution Neural Networks on the representations of the audio. This report explores the application of machine learning algorithms to identify and classify the genre of the given audio file. The features which will be extracted by sampling the audio into four random parts, thus augmenting the data into 8000 clips as a raw input file [1]. This raw data is also converted into mel-spectrograms. The features from both these input data are then fed to the Convolution Neural Network which are further trained to classify the given audio file.

Our approach consists of collecting the data, pre-processing it so that we can extract features such as time domain and frequency domain, building the model, training the classifier by feeding it with the data. For which, CNN based deep learning model will be implemented and compared to check which model works the best by using evaluation metrics on every model.

2. Literature Survey

[1] This is a follow up on the research paper on Music Genre Classification where the author(s) has implemented a variety of classification algorithms with different types of input. Here, the author has experimented with the RBF kernel support vector machine, k-nearest neighbours, a basic feed-forward network, and an advanced convolutional neural network. The dataset is obtained from GTZAN (same as ours) for all the musical data. The

input to the algorithms were raw amplitude data as well as transformed mel-spectrograms of that raw amplitude data. Mel-spectrogram represents an acoustic time-frequency representation of a sound. Then displayed an output of a predicted genre out of 10 common music genres. By converting the raw audio into mel-spectrograms, it produced better results on all the implemented models, with convolutional neural network showing better accuracy than the rest. In the results however, the models couldn't categorize pop music whilst retaining strong performance on classical music. As the pop music graphs are more scattered comparatively, this caused discrepancies in the results as there's lack of distinct style in case of pop music whereas classical music has a much clearer definition.

Related to our approach:

The data used here is same as what we are using. The pre-processing of the data and the approach to improving the performance by converting the raw data to mel-spectrogram is like the approach we are doing.

In [2], the author has widened the approach for automatic music genre detection and tagging using convolutional neural networks. The author is using the same dataset as us, which is GTZAN. Along with it, he is even using MagnaTagATune using the audio clips mel-spectrograms as input. For GTZAN, the dataset is split into 33% for validation and the rest for training. Training is done using Adam optimizer with learning rate of 0.001 and decay of 0.01. For MagnaTagATune, the dataset is split into 20% for test and 80% for training. 20% of the training dataset is used for validation. Training is done using Adam optimizer with an initial learning rate of 0.0001 and exponential learning rate decay. He calculated accuracy for the same with a decent accuracy as a result. Maybe the author didn't train for enough epochs, as he couldn't get accuracies more than 85% for either of the datasets.

Related to our approach:

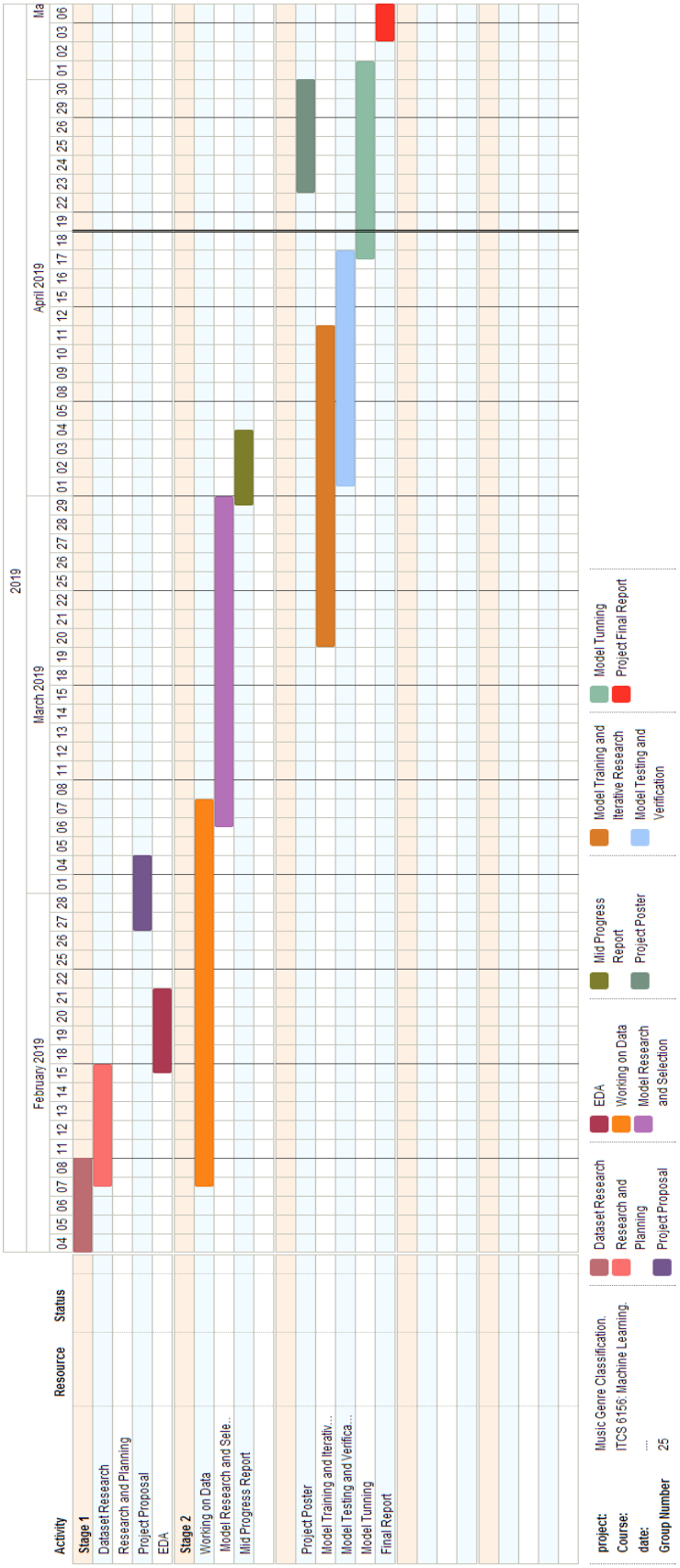
The data used in related works, is the same as what we have used. Rest other approaches are impressive and could be considered in the future; especially the Adam Optimizer methodology. But other than using CNN, K- Nearest Neighbours, nothing else is related to the approach we are using.

3. Difficulties or problems experienced

During processing the audio files, we faced problems with the huge memory consumption. Managing such big files, while implementing visualization also led to slower execution. The same problem was expected in future while model training. Hence, we decided to deal with it by reducing the sample time and still attempted to maintain the genre class uniqueness of audio.

Upon studying the visualization, we observed that most of the genres mel-spectrograms are hard to differentiate from. The dataset and its features are linearly inseparable. We are going to address this by using Support Vector Machines.

4. Modified plan / timeline



Accomplished milestones from original plan:

1. Requirement gathering and dataset collection
2. Pre-processing of data from .au format
3. Converting raw audio files to Short-Time Fourier Transform (STFT)
4. Visualization of data in mel-spectrogram

Team Members	**Contributions**
Mayuri Kingre	1. Data Collection 2. Pre-processing the dataset a. Data Reading b. Data Cleaning. 3. Implementing Support Vector Machine 4. Using evaluation metrics
Sayali More	1. Pre-processing the dataset a. Using Optimizers 2. Implementing K- Nearest Neighbour 3. Comparing the different model's accuracy
Nishita Kalyanpur	1. Pre-processing the dataset: a. Feature Extraction/ Feature selection b. Converting the raw input data to mel-spectrograms 2. Data Visualization and Analysis 3. Calculating accuracy 4. Analysing model performances using different performances metrics
Grishma Kalidindi	1. Selecting dataset. 2. Data Partitioning. 3. Implementing Convolution Neural Network. 4. Implementing Cross validation. 5. Using Encoders

5. Response to the feedback

Does the data have audio files and its genre label only?
The GTZAN dataset consists of audio files in .au format. Each genre is pre-classified in ten different folders. Besides which each file's title also contains its genre name. We get to utilize around 1000 song clips of 30 seconds duration.

I don't see any uniqueness.
We have tried to address this point by planning to approach additional methods like encoding. By encoding an audio, we expect to reveal different relations between the features of audio and its genres. We also plan on optimizing the dataset, by using an optimizer on dataset.

Data pre-processing: the data needs to be in an understandable format. What do you mean by understandable explain?
From the perspective of visualization, we are implementing mel-spectrograms. Mel-spectrograms are the acoustic time-frequency representations of a sound. These readings can also be used as input for CNN, by pre-processing using Fourier Transform as mentioned.

Why did you come up with this approach? Why does it make sense to you?
When we started with the audio files, we thought we could capture the most frequent frequency and use it as an input value for the neural network. Later we thought we could use the frequency graph like Mel-spectrograms as an image and give it as an input to a CNN.
And then after the literature survey, we found that the whole song is not required, we can sample the songs and then use these samples as inputs. Bringing together all our previous learnings, ideas and survey we came up with this approach.
Consequently, to bring novelty to our project we are thinking of using encoders.

Missing timeline
We have added a Gantt Chart as timeline in the Mid – Progress report.

6. Data and approaches:

The data chosen is from the GTZAN genre collection dataset. This dataset provides us with 1000 30-second audio clips, all labelled as one out of 10 possible genres and presented as .au files.

Link: http://marsyas.info/downloads/datasets.html

From each clip, sampling is done at four random locations, thus augmenting the data into 4000 clips of two seconds each. This leaves us with 44100 features for the raw audio input. Also, this raw audio file is converted into mel-spectrogram for performance

increase. Mel-spectrograms are a commonly used method of featuring audio as they closely represent how humans perceive audio (i.e. in log frequency). To convert files into mel-spectrogram, Fourier transform must be applied on the sampled raw input files.

The dataset consists of many folders containing audio files (having .au extensions) categorized into following music genres:

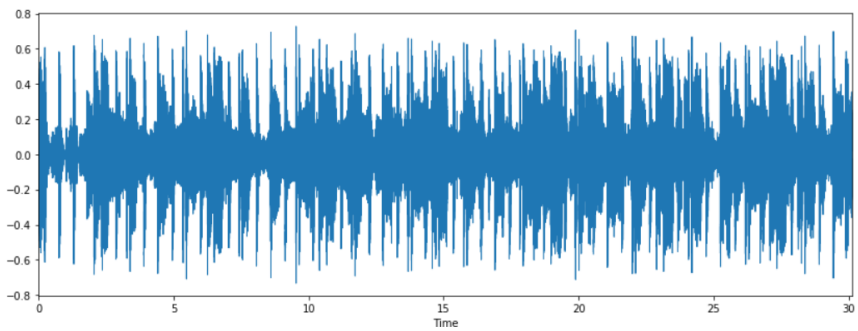
- Blues
- Classical
- Country
- Disco
- Hip-hop
- Jazz
- Metal
- Pop
- Reggae
- Rock

We start with processing the format of audio files. The original files are in the form of 30 seconds of audio clips in .au format. With consideration to huge memory usage and longer training time for models, we decide to split the audio files and use them in smaller parts. For processing the audio files, we split it into 2-seconds window frames in four random locations. The audio files are sampled at 22050 Hz, and this enables us to obtain around 44100 features for them [1].

Using the Librosa library, we implemented mel-spectrograms for visualizing the audio samples. Mel-spectrograms are the acoustic time-frequency representations of a sound. They help us to understand the relation in various frequency and amplitude combinations, and to study which combinations make a song associated with a genre. We have applied short-time Fourier Transform, to audio for enabling mel-spectrogram conversion.

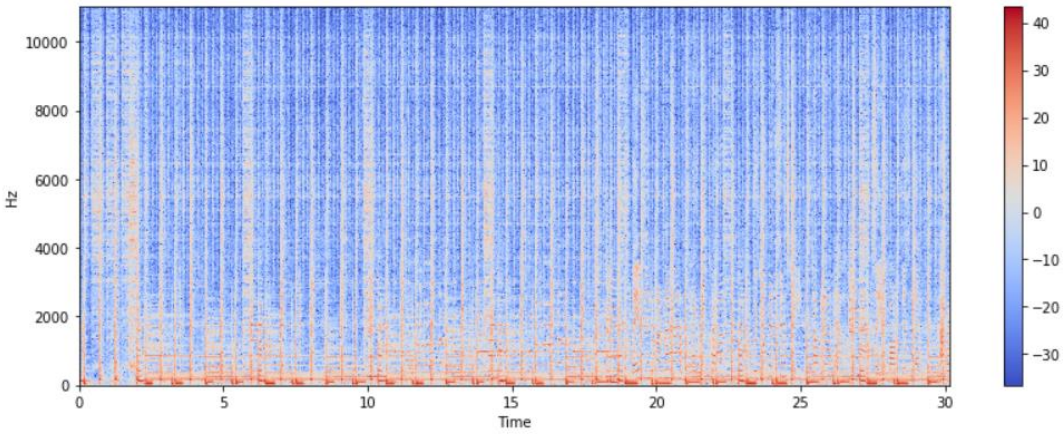
```
%matplotlib inline

import matplotlib.pyplot as plt
import librosa.display
plt.figure(figsize=(14, 5))
librosa.display.waveplot(x, sr=sr)
```

```
<matplotlib.collections.PolyCollection at 0x29beb761978>
```



```
X = librosa.stft(x)
Xdb = librosa.amplitude_to_db(abs(X))
plt.figure(figsize=(14, 5))
librosa.display.specshow(Xdb, sr=sr, x_axis='time', y_axis='hz')
plt.colorbar()
```

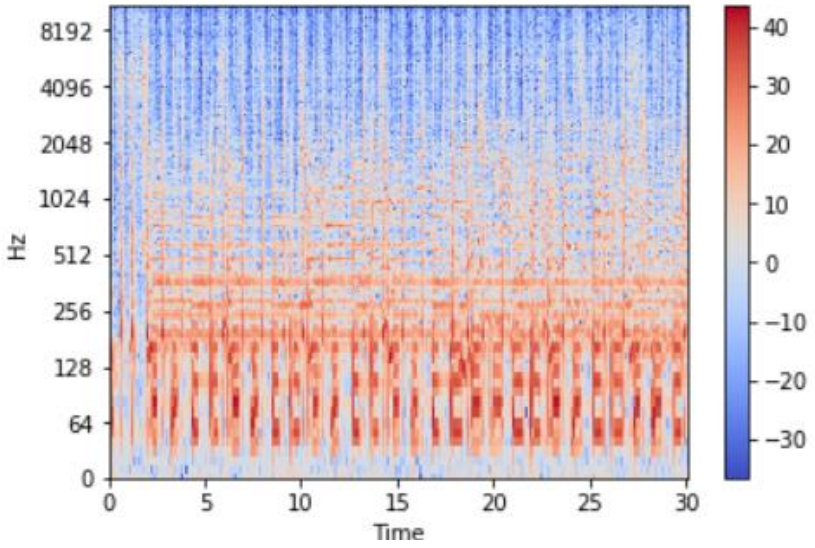
```
<matplotlib.colorbar.Colorbar at 0x29beb7ca588>
```



The vertical axis shows frequencies (from 0 to 10kHz), and the horizontal axis shows the time of the clip. Since we see that all action is taking place at the bottom of the spectrum, we can convert the frequency axis to a logarithmic one.

```
librosa.display.specshow(Xdb, sr=sr, x_axis='time', y_axis='log')
plt.colorbar()
```

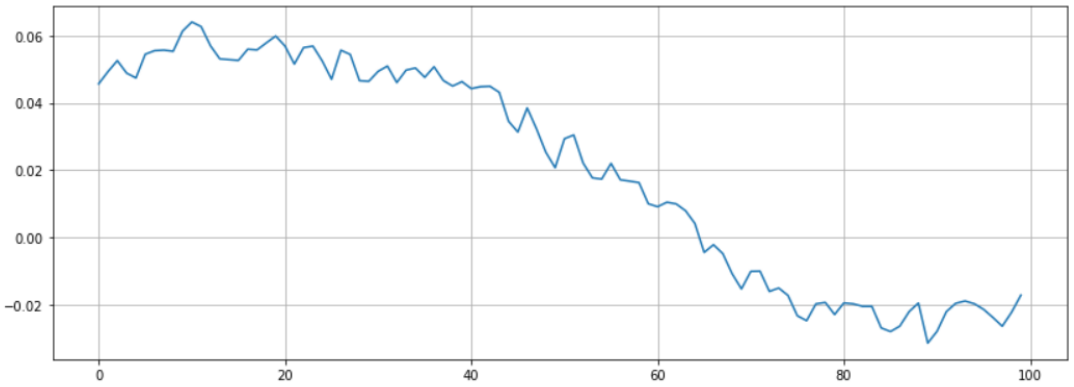
```
<matplotlib.colorbar.Colorbar at 0x29beb87ad30>
```



Looking for zero crossing rate:

This rate tells about the rate of sign-changes along a signal, i.e., the rate at which the signal changes from positive to negative or back. This feature has been used heavily in both speech recognition and music information retrieval.

```
n0 = 9000
n1 = 9100
plt.figure(figsize=(14, 5))
plt.plot(x[n0:n1])
plt.grid()
```

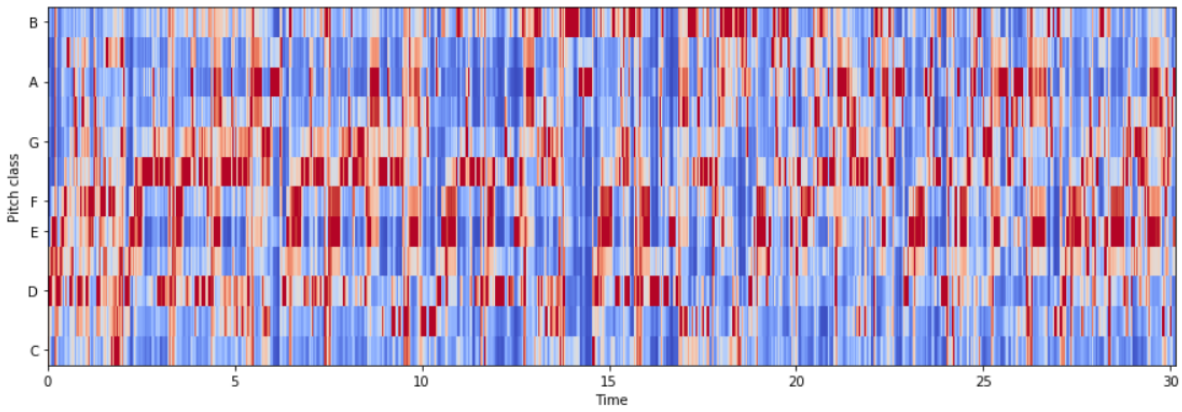


Chroma Frequencies:

Chroma features are used to project the entire spectrogram onto 12 bins representing the 12 distinct semitones (or chroma) of the musical octave.

```
# Loadign the file
x, sr = librosa.load('disco.00000.au')
hop_length = 512
chromagram = librosa.feature.chroma_stft(x, sr=sr, hop_length=hop_length)
plt.figure(figsize=(15, 5))
librosa.display.specshow(chromagram, x_axis='time', y_axis='chroma', hop_length=hop_length, cmap='coolwarm')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x29be8ae3400>
```



The above plot tells us about the Pitch classes and how the signal varies across every pitch.

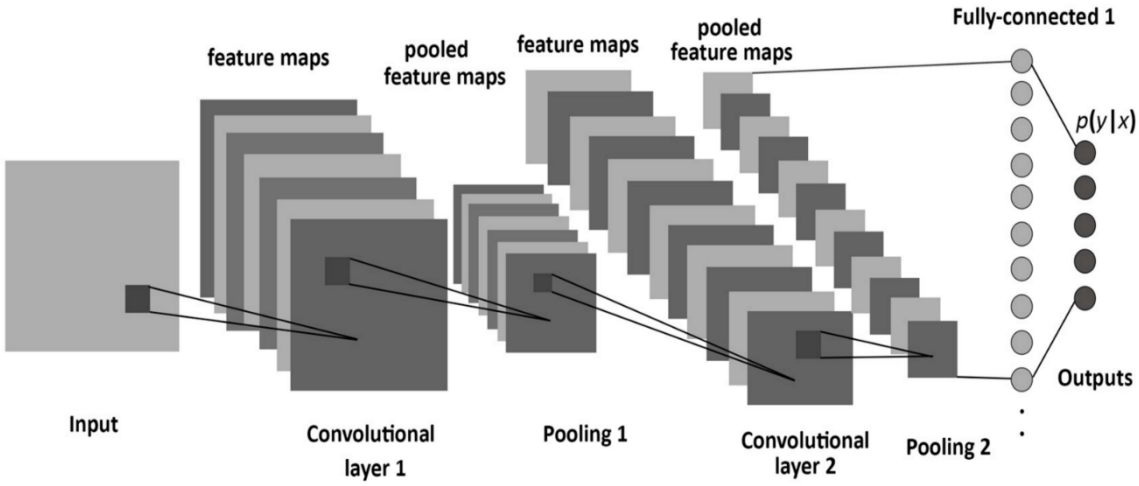
K- Nearest Neighbours – K Nearest Neighbour algorithm centres the classes of data by choosing the nearest data samples most relevant to the class. Using kNN we will segregate the music genre based on frequency and other aspects of music files.

Support Vector Machines (SVM) – This is a supervised learning algorithm, which can be used for both regression and classification. It finds a hyperplane in given features and distinctly classifies the data.

This is one of the popular optimal margin classifiers, which uses kernels to find hidden relations in data. Our data is linearly inseparable, which is addressable with SVM.

Convolution Neural Network (CNN) – This is a deep learning algorithm which, based on features of input data, generates weights and biases, to learn from it. It uses multilayer perceptrons and max pools. Max pooling is a sample-based discretization process. Its objective is to down-sample an input representation and reduce over-fitting by presenting it in an abstracted format.

There is relatively minimum data pre-processing needed for it. In each layer, data would go through regularization and an activation function (ReLU, softmax, etc). This will be implemented using TensorFlow and Keras.



Keras Sequential Model – Sequential model is a linear stack of layers. We can define different layers for the model. The input parameters to the model like Optimizers (to control gradient clipping), activation functions (different for different layers), Loss function (the objective that the model tries to minimize) are used while training the data.

7. Difference / Novelty

We are also considering using auto-encodings with neural networks for learning genre embeddings, in order to get and analyse different output. Here we would use hidden layers to implement audio encoders and reconstruct them after getting predictions. By encoding an audio, we expect to reveal different relations between the features of audio and its genres.
We plan to implement this and try to attempt genre classification with k-NN or some other unsupervised model. Keras Sequential Model will also be implemented. Its one of the models used for convolution/ recurrent networks.

8. References

[1] Huang, Derek A., Serafini Arianna A., Pugh Eli J., et al. “Music Genre Classification”. 2018.

http://cs229.stanford.edu/proj2018/report/21.pdf

[2] Flores, Miguel, et al. “Deep Music Genre”. 2017

http://cs231n.stanford.edu/reports/2017/pdfs/22.pdf

[3] G. Tzanetakis, P. Cook, et al. “Musical genre classification of audio signals”, IEEE, 2002

https://ieeexplore.ieee.org/document/1021072

[4] Michael Haggblade, Yang Hong, Kenny Kao, et al. “Music Genre Classification”. 2011.

http://cs229.stanford.edu/proj2011/HaggbladeHongKao-MusicGenreClassification.pdf