In [1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [2]:
```
from keras.models import load_model
model = load_model('/content/m01.h5')
```

In [4]:
```
# Usual Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import os
import matplotlib.pyplot as plt
%matplotlib inline
import sklearn

# Librosa (the mother of audio files)
import librosa
import librosa.display
import IPython.display as ipd
import warnings
warnings.filterwarnings('ignore')
```

In [5]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 510, 256)          25600

 max_pooling1d (MaxPooling1  (None, 255, 256)          0
 D)

 conv1d_1 (Conv1D)           (None, 253, 128)          98432

 max_pooling1d_1 (MaxPoolin  (None, 126, 128)          0
 g1D)

 lstm (LSTM)                 (None, 126, 64)           49408

 lstm_1 (LSTM)               (None, 32)                12416

 dropout (Dropout)           (None, 32)                0

 dense (Dense)               (None, 10)                330

=================================================================
Total params: 186186 (727.29 KB)
Trainable params: 186186 (727.29 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [6]:

```
from keras.models import load_model
model = load_model('/content/m01.h5')
```

In [7]:

```
genres_dir = "/content/drive/MyDrive/genres_original"
data = np.zeros((999,512,33), dtype=np.float64)
target=[]
# List of genre names
genre_names = ["blues", "classical", "country", "disco", "hiphop", "jazz'
i=0
for genre in genre_names:
    genre_path = os.path.join(genres_dir, genre)

    # Loop through each file in the genre folder
    for filename in os.listdir(genre_path):
        file_path = os.path.join(genre_path, filename)
        y, sr = librosa.load(file_path)
        y, _ = librosa.effects.trim(y)
        mfcc = librosa.feature.mfcc(y=y, sr=sr, hop_length=512, n_mfcc=1
        spectral_center = librosa.feature.spectral_centroid(y=y, sr=sr, 
        chroma = librosa.feature.chroma_stft(y=y, sr=sr, hop_length=512)
        spectral_contrast = librosa.feature.spectral_contrast(y=y, sr=sr
        target.append(genre)
        data[i, :, 0:13] = mfcc.T[0:512,:]
        data[i, :, 13:14] = spectral_center.T[0:512, :]
        data[i, :, 14:26] = chroma.T[0:512, :]
        data[i, :, 26:33] = spectral_contrast.T[0:512, :]
        print("Numerical features extracted from audio file %i of %i." %
        i+=1
```

```
Numerical features extracted from audio file 1 of 999.
Numerical features extracted from audio file 2 of 999.
Numerical features extracted from audio file 3 of 999.
Numerical features extracted from audio file 4 of 999.
Numerical features extracted from audio file 5 of 999.
Numerical features extracted from audio file 6 of 999.
Numerical features extracted from audio file 7 of 999.
Numerical features extracted from audio file 8 of 999.
Numerical features extracted from audio file 9 of 999.
Numerical features extracted from audio file 10 of 999.
Numerical features extracted from audio file 11 of 999.
Numerical features extracted from audio file 12 of 999.
Numerical features extracted from audio file 13 of 999.
Numerical features extracted from audio file 14 of 999.
Numerical features extracted from audio file 15 of 999.
Numerical features extracted from audio file 16 of 999.
Numerical features extracted from audio file 17 of 999.
Numerical features extracted from audio file 18 of 999.
Numerical features extracted from audio file 19 of 999.
Numerical features extracted from audio file 20 of 999.
```

In [8]:

```
y=np.zeros((999,10))
for i,genre in enumerate(target):
  ind=genre_names.index(genre)
  y[i,ind]=1
```

In [9]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(data, y, test_size=0
```

In [10]:

```
import math
# score, accuracy = model.evaluate(
#     x_test, y_test, batch_size=batch_size, verbose=1
# )
batch_size = 35  # num of training examples per minibatch
num_epochs = 400
num_batches = math.ceil(len(x_test) // batch_size)
accuracies = []

for i in range(num_batches):
    start = i * batch_size
    end = (i + 1) * batch_size
    batch_x = x_test[start:end]
    batch_y = y_test[start:end]
    _, batch_accuracy = model.evaluate(batch_x, batch_y, verbose=0)
    accuracies.append(batch_accuracy)

mean_test_accuracy = round(np.mean(accuracies),4)
print("Accuracy of GTZAN Test Dataset:", mean_test_accuracy)
```

```
Accuracy of GTZAN Test Dataset: 0.5347
```

In [11]:

```
genres_dir = "/content/drive/MyDrive/songs_final"
val_data = np.zeros((100,512,33), dtype=np.float64)
target=[]
# List of genre names
genre_names = ["blues", "classical", "country", "disco", "hiphop", "jazz'
i=0
for genre in genre_names:
    genre_path = os.path.join(genres_dir, genre)

    # Loop through each file in the genre folder
    for filename in os.listdir(genre_path):
        file_path = os.path.join(genre_path, filename)
        y, sr = librosa.load(file_path)
        y, _ = librosa.effects.trim(y)
        mfcc = librosa.feature.mfcc(y=y, sr=sr, hop_length=512, n_mfcc=1
        spectral_center = librosa.feature.spectral_centroid(y=y, sr=sr,
        chroma = librosa.feature.chroma_stft(y=y, sr=sr, hop_length=512)
        spectral_contrast = librosa.feature.spectral_contrast(y=y, sr=sr
        target.append(genre)
        print(file_path)
        val_data[i, :, 0:13] = mfcc.T[0:512,:]
        val_data[i, :, 13:14] = spectral_center.T[0:512, :]
        val_data[i, :, 14:26] = chroma.T[0:512, :]
        val_data[i, :, 26:33] = spectral_contrast.T[0:512, :]
        print("Numerical features extracted from audio file %i of %i." %
        i+=1
```

```
/content/drive/MyDrive/songs_final/blues/blues2.wav
Numerical features extracted from audio file 1 of 100.
/content/drive/MyDrive/songs_final/blues/blues1.wav
Numerical features extracted from audio file 2 of 100.
/content/drive/MyDrive/songs_final/blues/blues3.wav
Numerical features extracted from audio file 3 of 100.
/content/drive/MyDrive/songs_final/blues/blues4.wav
Numerical features extracted from audio file 4 of 100.
/content/drive/MyDrive/songs_final/blues/blues6.wav
Numerical features extracted from audio file 5 of 100.
/content/drive/MyDrive/songs_final/blues/blues5.wav
Numerical features extracted from audio file 6 of 100.
/content/drive/MyDrive/songs_final/blues/blues7.wav
Numerical features extracted from audio file 7 of 100.
/content/drive/MyDrive/songs_final/blues/blues8.wav
Numerical features extracted from audio file 8 of 100.
/content/drive/MyDrive/songs_final/blues/blues9.wav
Numerical features extracted from audio file 9 of 100.
/content/drive/MyDrive/songs_final/blues/blues10.wav
Numerical features extracted from audio file 10 of 100.
```

In [12]:

```
val_y=np.zeros((100,10))
for i,genre in enumerate(target):
  ind=genre_names.index(genre)
  val_y[i,ind]=1
```

```
In [14]: import math
         # score, accuracy = model.evaluate(
         #     x_test, y_test, batch_size=batch_size, verbose=1
         # )

         num_batches = math.ceil(len(val_data) // batch_size)
         accuracies = []

         for i in range(num_batches):
             start = i * batch_size
             end = (i + 1) * batch_size
             batch_x = val_data[start:end]
             batch_y = val_y[start:end]
             _, batch_accuracy = model.evaluate(batch_x, batch_y, verbose=0)
             accuracies.append(batch_accuracy)

         mean_test_accuracy = round(np.mean(accuracies),4)
         print("Accuracy on newly collected data:", mean_test_accuracy)
```

```
Accuracy on newly collected data: 0.3143
```