# Music Genre Classification

Springboard DSC Program
Capstone Project 2
July 2020

By Morgan Fry

# Introduction

## ***The Problem:***

Classifying songs by genre is useful for library management and recommendation engines. It can be labor intensive.

## ***The Client:***

Music streaming services like Spotify, Apple Music, etc. use recommendation engines to better serve new music to users. Classifying accurately by features in the music might improve these services.

## ***The Goal:***

Sufficiently accurate genre classification based on audio features.

# Data Science Problem and the data

This is a supervised learning problem, a multiclass classification problem with one of 8 labels assigned to each sample.

The data used is the Free Music Archive, a collection of music samples made available for machine learning.

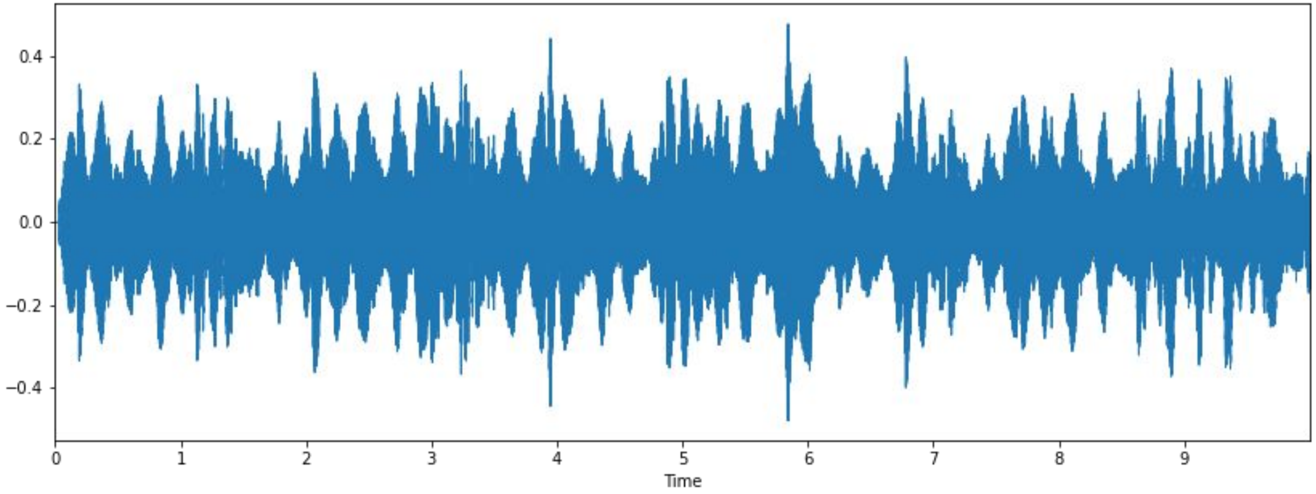
# Data Wrangling and Acquisition

Data from FMA is already well organized.

- 100,000 30-second clips
- Already split into train, validation, test groups
- Decoded with Tensorflow TFIO tools
- A few (164) corrupted files
- Using an 8000 sample subset with 8 classes

# Exploratory Data Analysis (EDA)

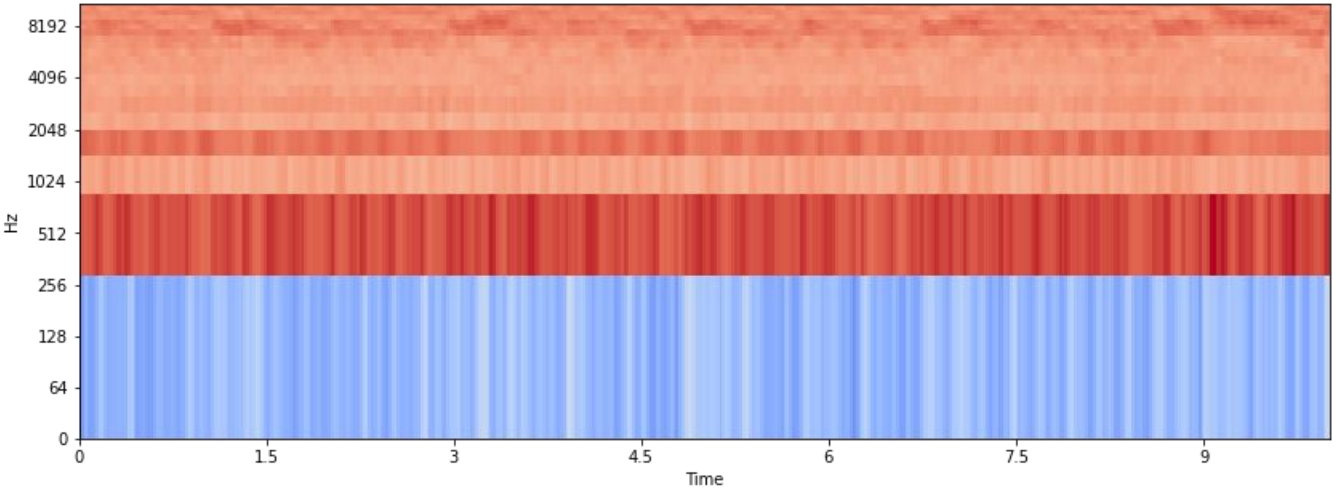
Using mp3 files -- when decompressed, the plot of the a .wav file:



# EDA (cont.)

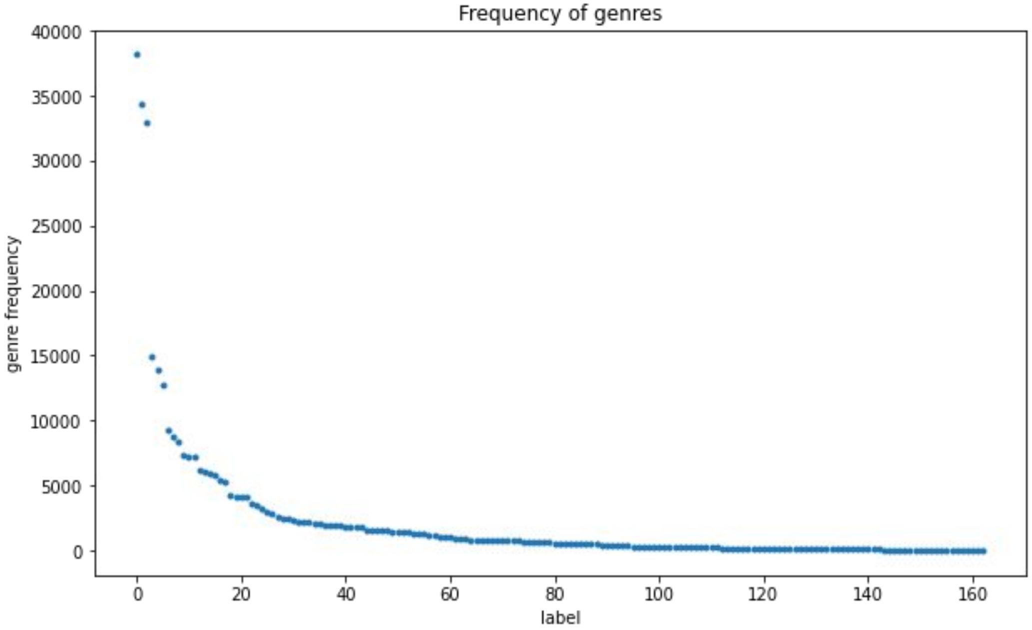
Relevant features for extraction:

MFCC (indication of timbre of audio)



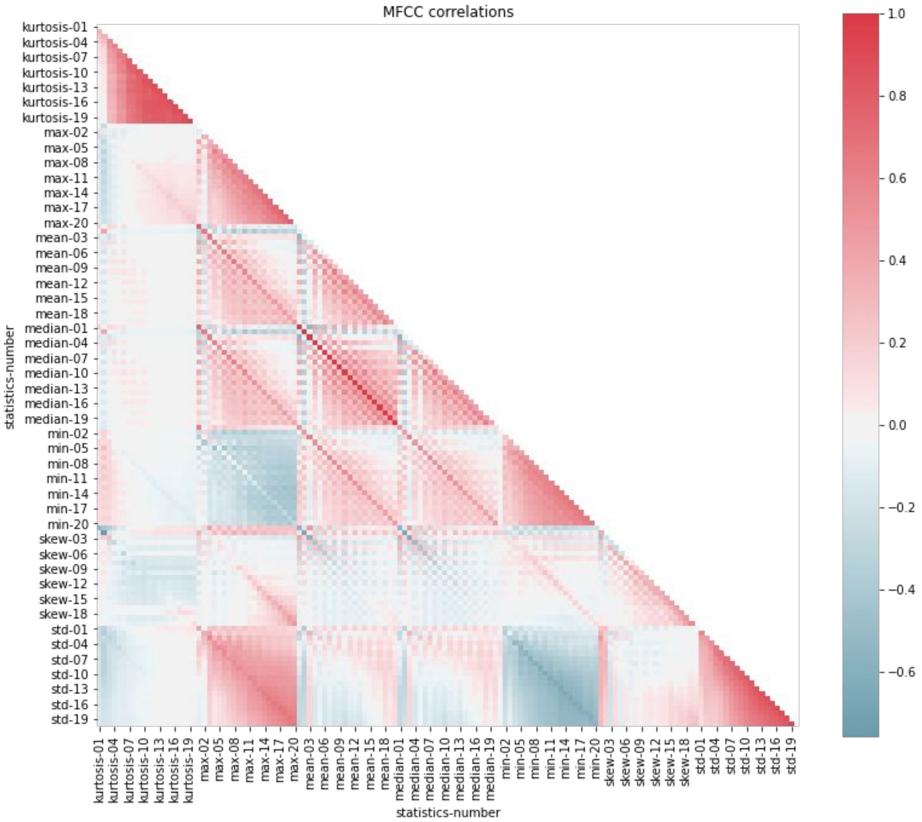
# EDA (cont.)

Distribution of all genres (classes) over entire dataset:



# EDA (cont.)

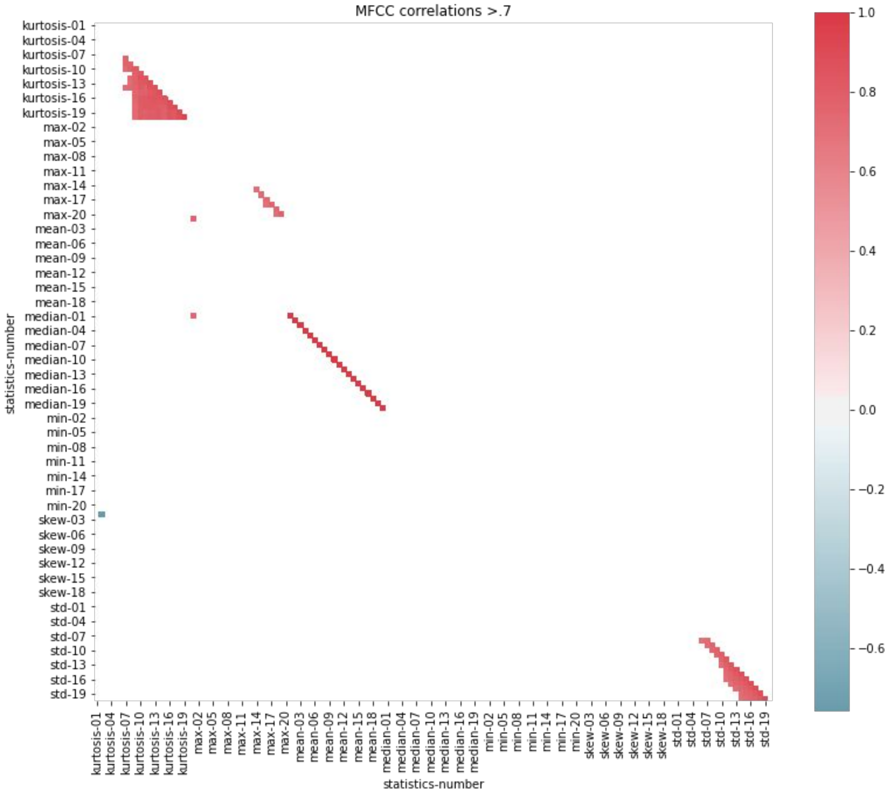
Correlation of features (MFCC)



# EDA (cont.)

Isolating very high correlations:

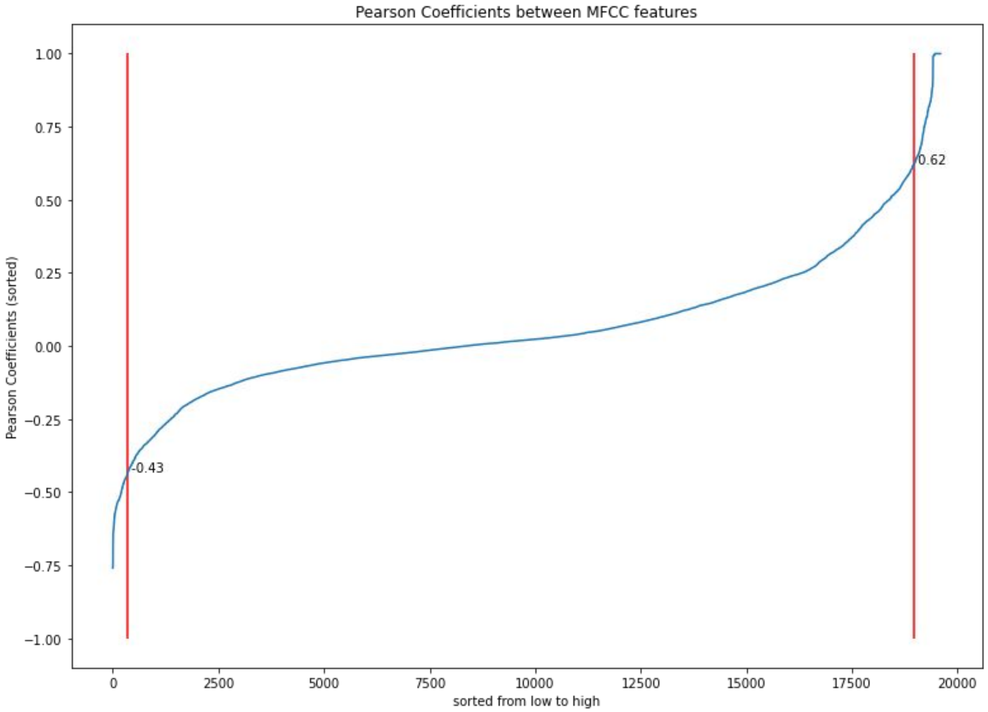
- Not many highly correlated pairs
- Highly correlated pairs limited to near neighbors in time



# EDA (cont.)

Pearson Coefficients between MFCC features:

- Few highly correlated pairs
- 95% of correlations between -0.43 and 0.62



# Baseline Model

## Logistic regression:

Classification Report (test set)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.26 | 0.35 | 0.30 | 96 |
| 1 | 0.15 | 0.21 | 0.18 | 98 |
| 2 | 0.15 | 0.13 | 0.14 | 100 |
| 3 | 0.32 | 0.19 | 0.24 | 100 |
| 4 | 0.24 | 0.25 | 0.25 | 100 |
| 5 | 0.23 | 0.16 | 0.19 | 100 |
| 6 | 0.08 | 0.08 | 0.08 | 100 |
| 7 | 0.30 | 0.34 | 0.32 | 100 |
| accuracy | | | 0.21 | 794 |
| macro avg | 0.22 | 0.21 | 0.21 | 794 |
| weighted avg | 0.22 | 0.21 | 0.21 | 794 |

# Baseline Model (cont.)

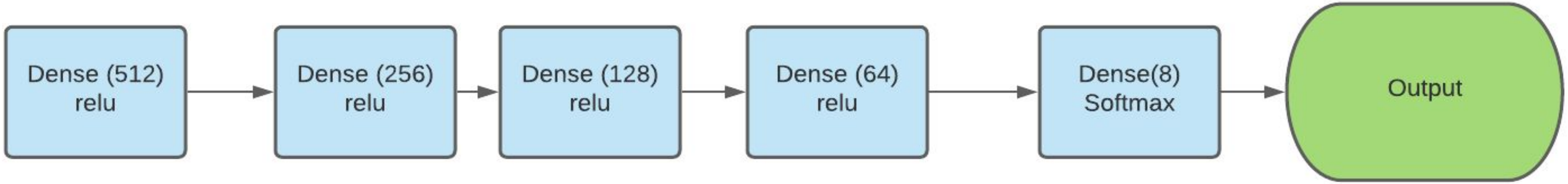
## Fully Connected Neural Net results:

Classification Report (test set)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.34 | 0.50 | 0.40 | 96 |
| 1 | 0.19 | 0.14 | 0.16 | 98 |
| 2 | 0.19 | 0.13 | 0.15 | 100 |
| 3 | 0.47 | 0.38 | 0.42 | 100 |
| 4 | 0.30 | 0.39 | 0.34 | 100 |
| 5 | 0.34 | 0.33 | 0.34 | 100 |
| 6 | 0.18 | 0.19 | 0.19 | 100 |
| 7 | 0.42 | 0.41 | 0.41 | 100 |
| accuracy | | | 0.31 | 794 |
| macro avg | 0.30 | 0.31 | 0.30 | 794 |
| weighted avg | 0.30 | 0.31 | 0.30 | 794 |

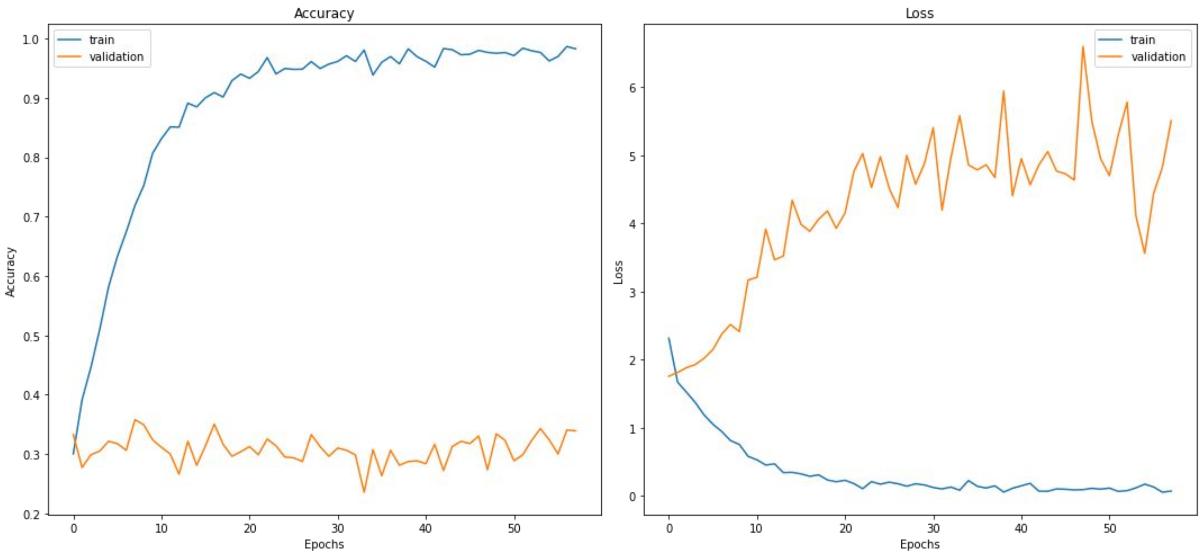
# Baseline Model (cont.)

Fully connected Neural Net architecture



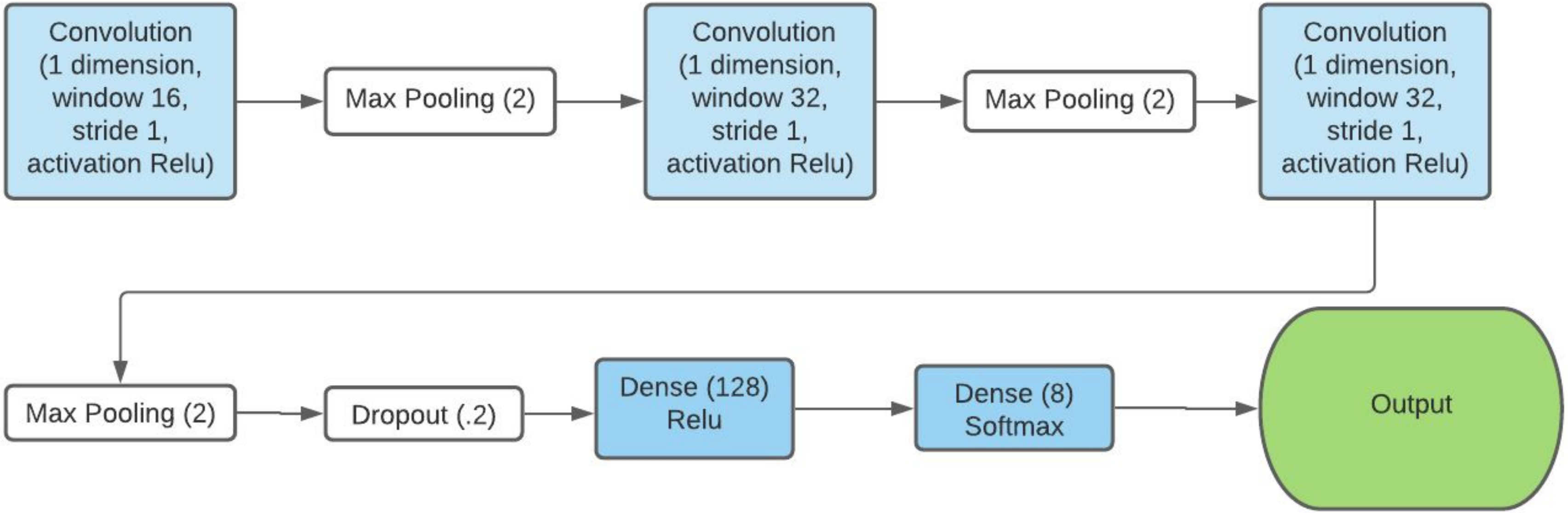
# Baseline Model (cont.)

Neural Net training:



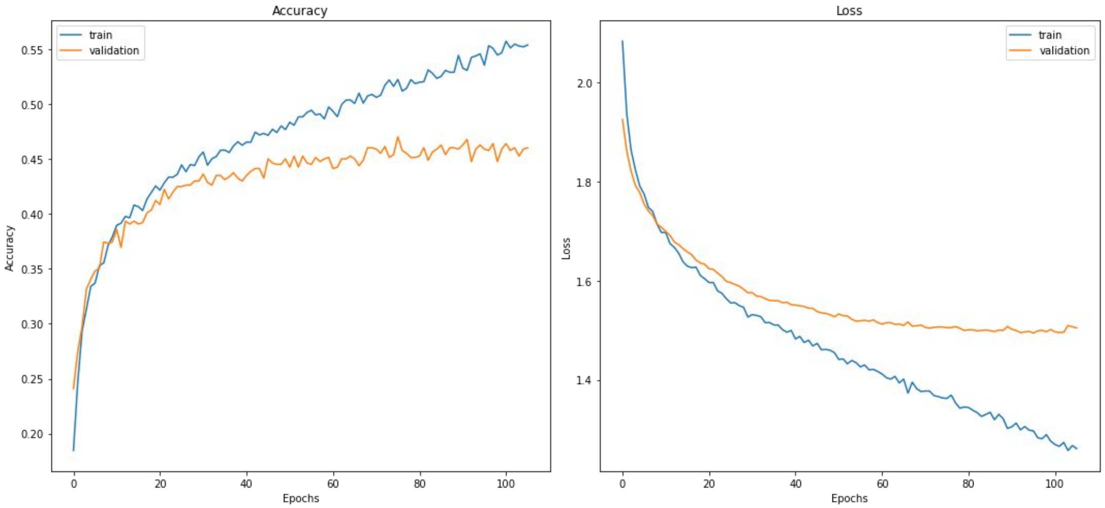
# Convolutional Neural Net (CNN)

CNN Architecture:



# CNN (cont.)

## CNN Training:



# CNN (cont.)

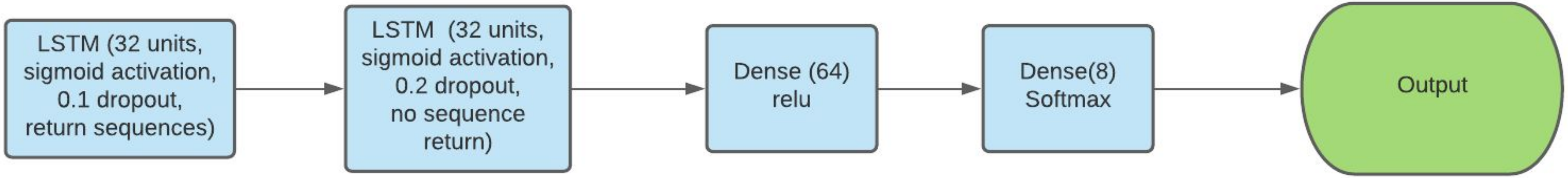
## CNN Results:

Classification Report (test set)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.47 | 0.42 | 0.44 | 96 |
| 1 | 0.13 | 0.10 | 0.11 | 98 |
| 2 | 0.18 | 0.24 | 0.21 | 100 |
| 3 | 0.49 | 0.71 | 0.58 | 100 |
| 4 | 0.33 | 0.40 | 0.36 | 100 |
| 5 | 0.41 | 0.36 | 0.38 | 100 |
| 6 | 0.17 | 0.07 | 0.10 | 100 |
| 7 | 0.52 | 0.56 | 0.54 | 100 |
| accuracy | | | 0.36 | 794 |
| macro avg | 0.34 | 0.36 | 0.34 | 794 |
| weighted avg | 0.34 | 0.36 | 0.34 | 794 |

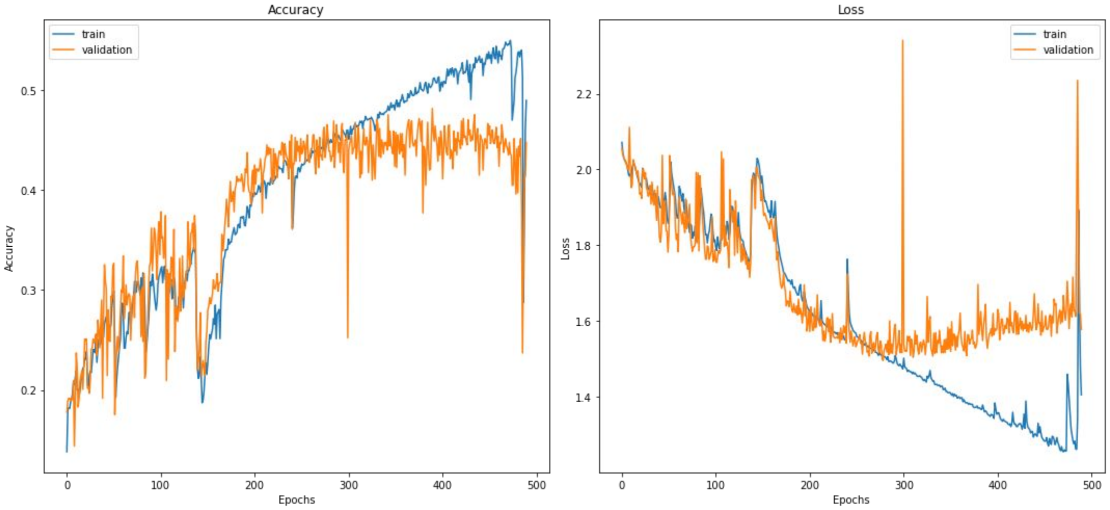
# Long Short Term Memory (LSTM)

LSTM Architecture:



# LSTM (cont.)

LSTM Training:



# LSTM (cont.)

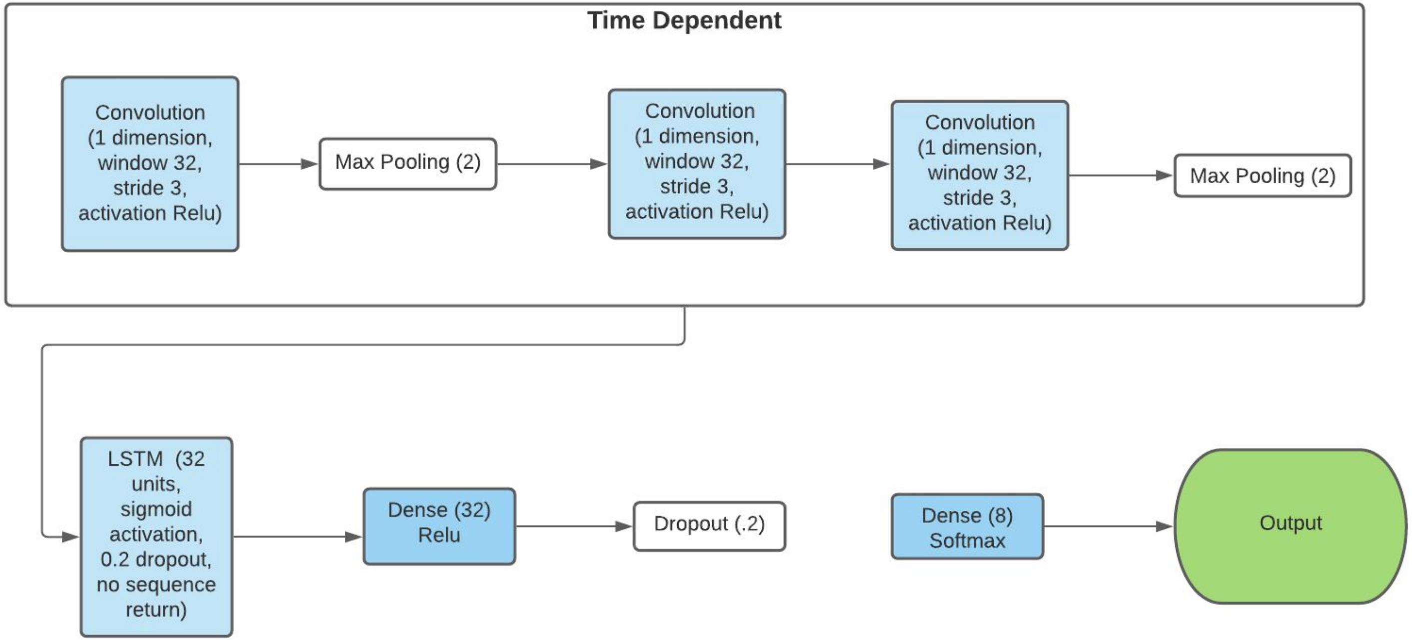
## LSTM Results:

Classification Report (test set)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.50 | 0.27 | 0.35 | 96 |
| 1 | 0.26 | 0.17 | 0.21 | 98 |
| 2 | 0.22 | 0.23 | 0.23 | 100 |
| 3 | 0.53 | 0.61 | 0.56 | 100 |
| 4 | 0.33 | 0.33 | 0.33 | 100 |
| 5 | 0.36 | 0.28 | 0.32 | 100 |
| 6 | 0.27 | 0.31 | 0.29 | 100 |
| 7 | 0.34 | 0.57 | 0.43 | 100 |
| accuracy | | | 0.35 | 794 |
| macro avg | 0.35 | 0.35 | 0.34 | 794 |
| weighted avg | 0.35 | 0.35 | 0.34 | 794 |

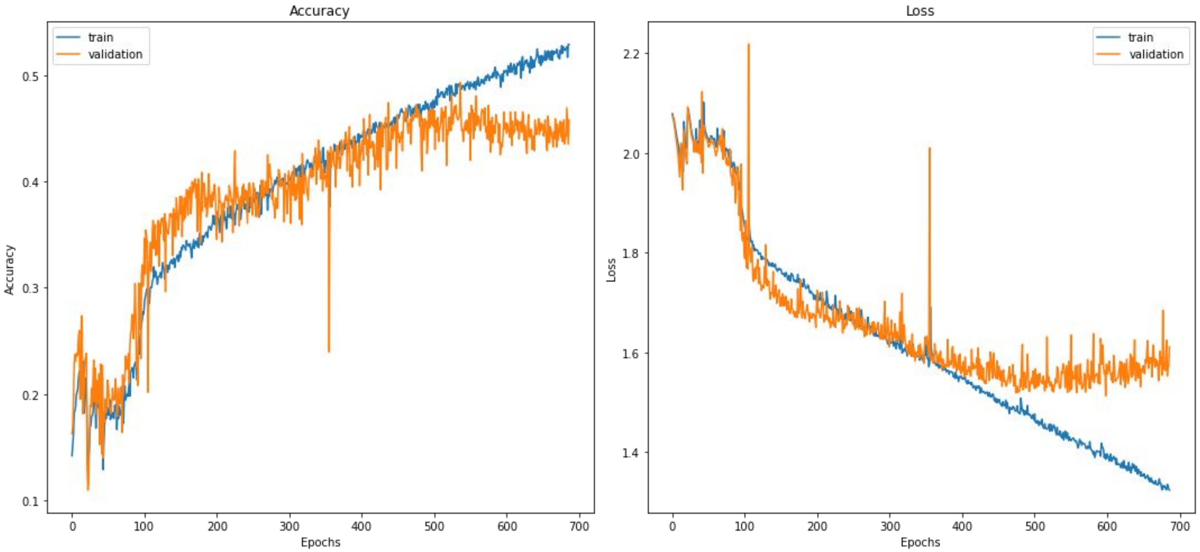
# Time-Dependent CNN (TD-CNN)

TD-CNN Architecture:



# TD-C-N (cont.)

TD-CNN Training:



# TD-CNN (cont.)

## TD-CNN Results:

Classification Report (test set)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.42 | 0.51 | 0.46 | 96 |
| 1 | 0.32 | 0.11 | 0.17 | 98 |
| 2 | 0.34 | 0.48 | 0.40 | 100 |
| 3 | 0.63 | 0.66 | 0.64 | 100 |
| 4 | 0.38 | 0.47 | 0.42 | 100 |
| 5 | 0.42 | 0.33 | 0.37 | 100 |
| 6 | 0.34 | 0.31 | 0.32 | 100 |
| 7 | 0.49 | 0.49 | 0.49 | 100 |
| accuracy | | | 0.42 | 794 |
| macro avg | 0.42 | 0.42 | 0.41 | 794 |
| weighted avg | 0.42 | 0.42 | 0.41 | 794 |

# Overall Results

| **Model** | **Accuracy(all classes)** | **Train Time** |
|---|---|---|
| Baseline | .30 | 5m |
| CNN | .36 | 3m |
| LSTM | .35 | 90m |
| TD-CNN | .42 | 3h 20m |

# Results (cont.)

| Class | Genre | F1 (CNN) | F1(LSTM) | F1(TD-CNN) |
|---|---|---|---|---|
| 0 | Electronic | .45 | .35 | .46 |
| 1 | Experimental | .18 | .21 | .17 |
| 2 | Folk | .19 | .23 | .40 |
| 3 | Hip-Hop | .66 | .56 | .64 |
| 4 | Instrumental | .33 | .33 | .42 |
| 5 | International | .44 | .32 | .37 |
| 6 | Pop | .22 | .29 | .32 |
| 7 | Rock | .52 | .43 | .49 |

# Conclusions

- TD-CNN model is most accurate (44%) but slowest (70m)
- CNN model is fastest (3m), with 36% accuracy
- LSTM model is slower than CNN and less accurate
- All models strong and weak on the same classes

# Recommendations

- Further development recommended, 44% accuracy means too many misclassifications to be useful
- Feature selection:
    - MFCCs are a good proxy for timbre but miss other musical features
    - Append features (e.g. tempograms) that capture rhythm-related musical features
- Train on larger dataset. We used 8000 of 106000 samples here.