UrbanSound8K Dataset Classification

CSML 1020 - Winter 2020 - Group 5

Dr. Karthik Kuber

Alex Fung, Patrick Osborne, Tony Lee, Viswesh Krishnamurthy

YORK UNIVERSITÉ UNIVERSITY

PROBLEM SELECTION & DEFINITION

- We chose to solve a classification problem with the UrbanSound8K Dataset

- The goal of this project was to develop a classifier model that can accept an audio file as input and classify the audio into one of the following 10 categories
  - Air conditioner (Class 0)
  - Car horn (1)
  - Children playing (2)
  - Dog bark (3)
  - Drilling (4)
  - Engine idling (5)
  - Gun shot (6)
  - Jackhammer (7)
  - Siren (8)
  - Street music (9)

RATIONALE

- We selected this problem and dataset to three specific goals

- 1. Allow us to contribute to the field
  - Audio classification is under-represented
  - Specifically sound classification – very positive results are rare and not trivial

- 2. Solve the problem through multiple different approaches
  - Generate numeral/vector features
  - Generate image-based features and run through image recognizers
  - Integrate both traditional ML models as well as Deep Learning (CNNs)

- 3. Real World Applicability
  - Classifying ambient, urban sounds has multiple practical uses
  - Government applications (data collection/urban planning), maintenance/manufacturing (identifying sounds of defects i.e. railway), sound/noise cancellation (i.e. Zoom)

BACKGROUND OF THE DATASET

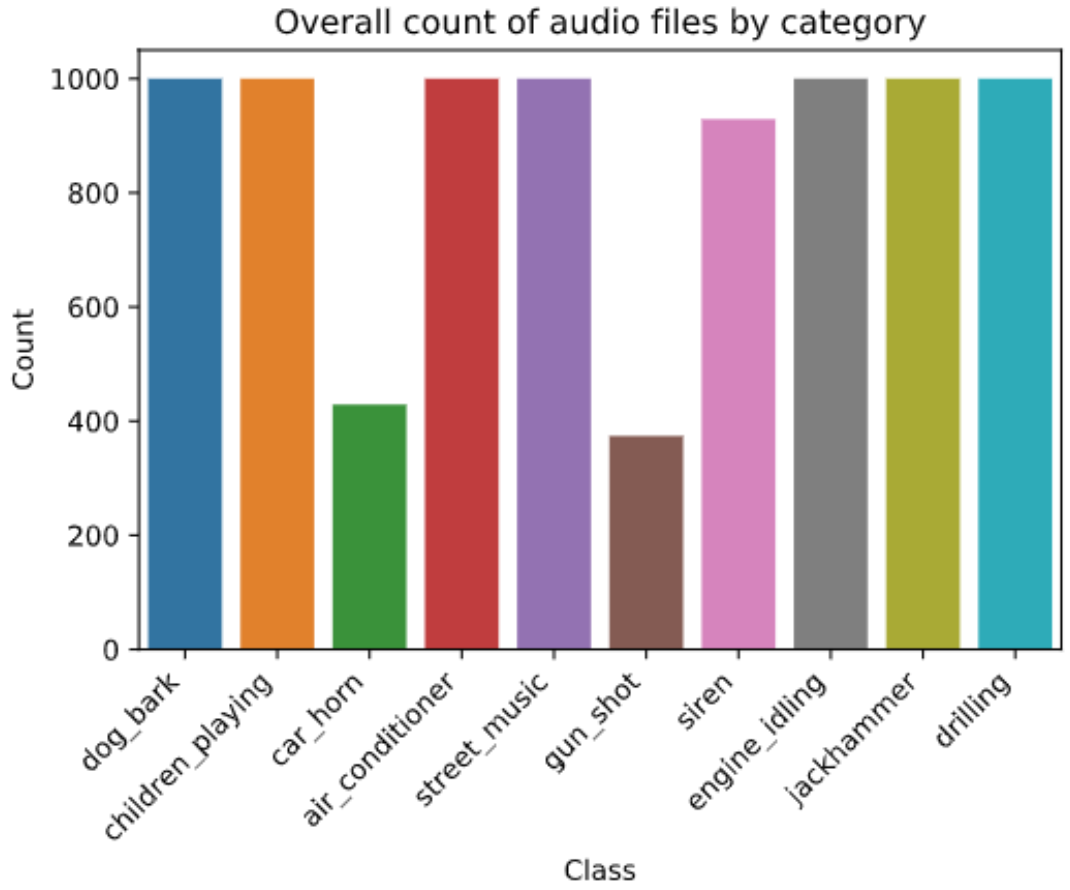
- UrbanSound8K is a dataset which contains a collection of 8732 labeled sound excerpts (<=4s) of urban sounds from 10 classes: air conditioner operation sounds, car horns, children playing, dog barks, drilling sounds, engine idling noises, gun shots, jackhammer operation sounds, sirens, and street music

- Furthermore, the dataset is divided into 10 folds, which is the standard by which the creators intend to keep the comparison of the performance of machine learning models easy and fair.

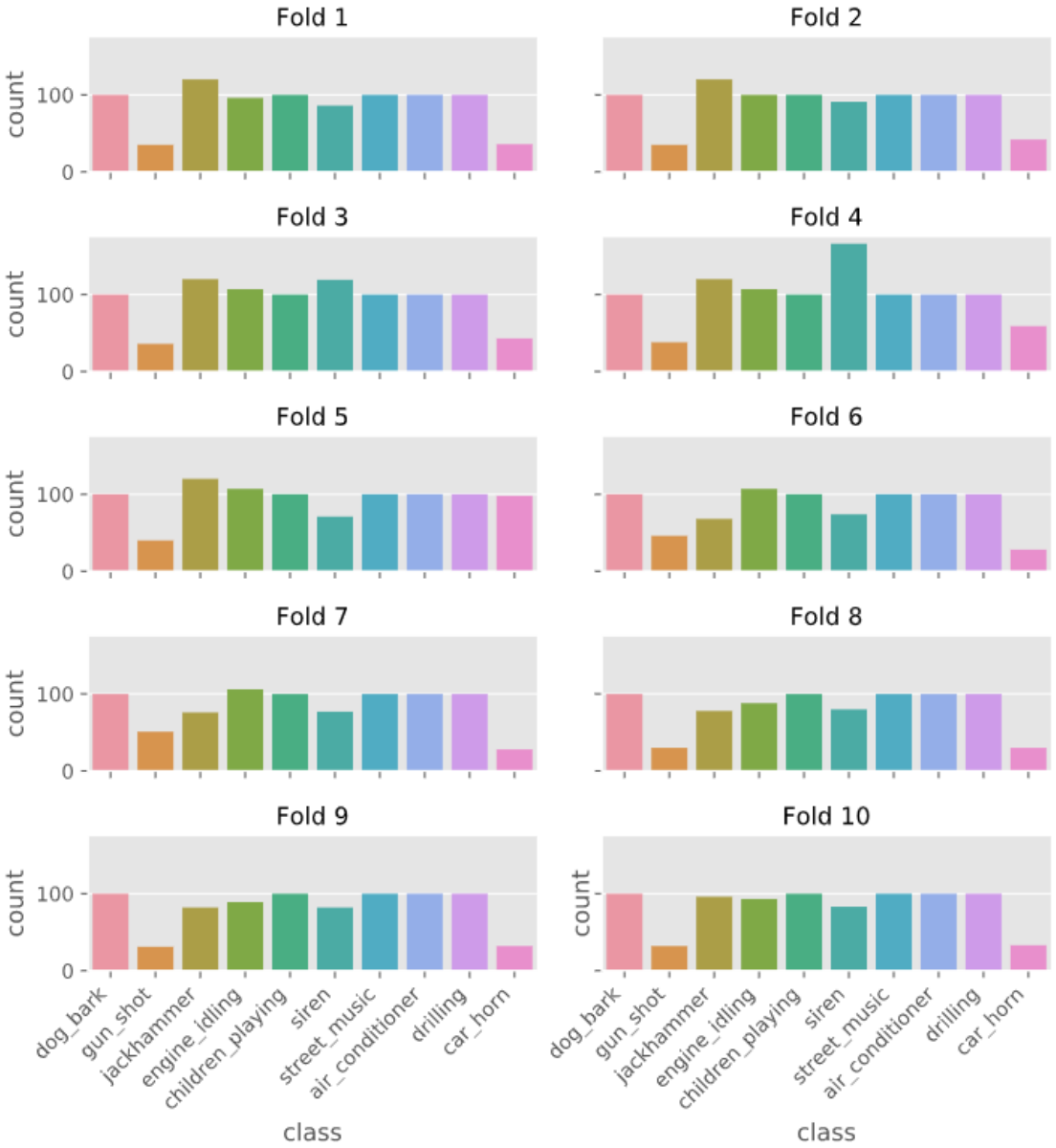
- They are in WAV format. According to the creators, all excerpts are taken from field recordings uploaded to www.freesound.org

DATA EXPLORATION

- As seen in the Figure, the 10 categories of sounds are mostly evenly distributed when viewed as a whole. Notably, audio samples of gun shots and car horns is about half as numerous as the other



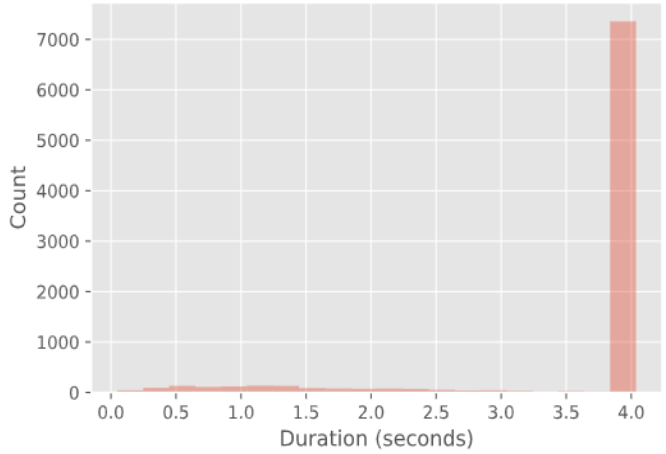
DATA EXPLORATION



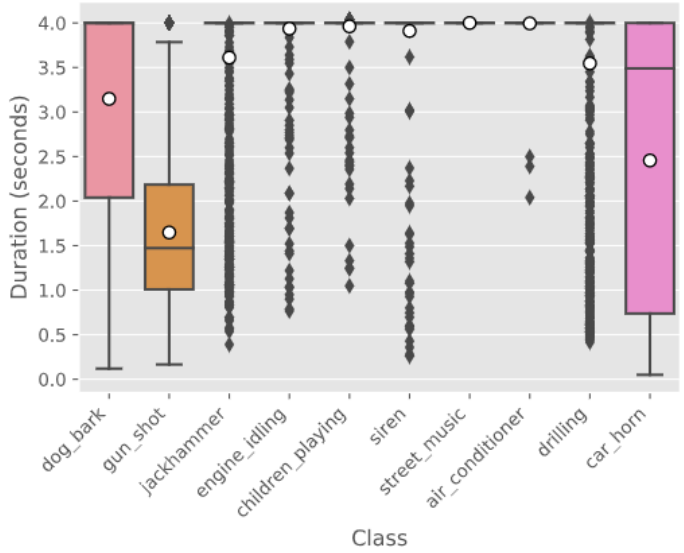
- This figure shows that the creators have done a good job of sampling down the data for each fold while still maintaining the distribution of the counts of each category. We see this when breakdown the counts of the categories by fold

DATA PREPROCESSING

- The histogram of duration of the audio files show that most of the audio files are 4 secs long



- However, the box plot confirmed that there is considerable variation in audio length between different classes. With this learning all audio files shorter than 4 secs, were looped to become 4 secs in length

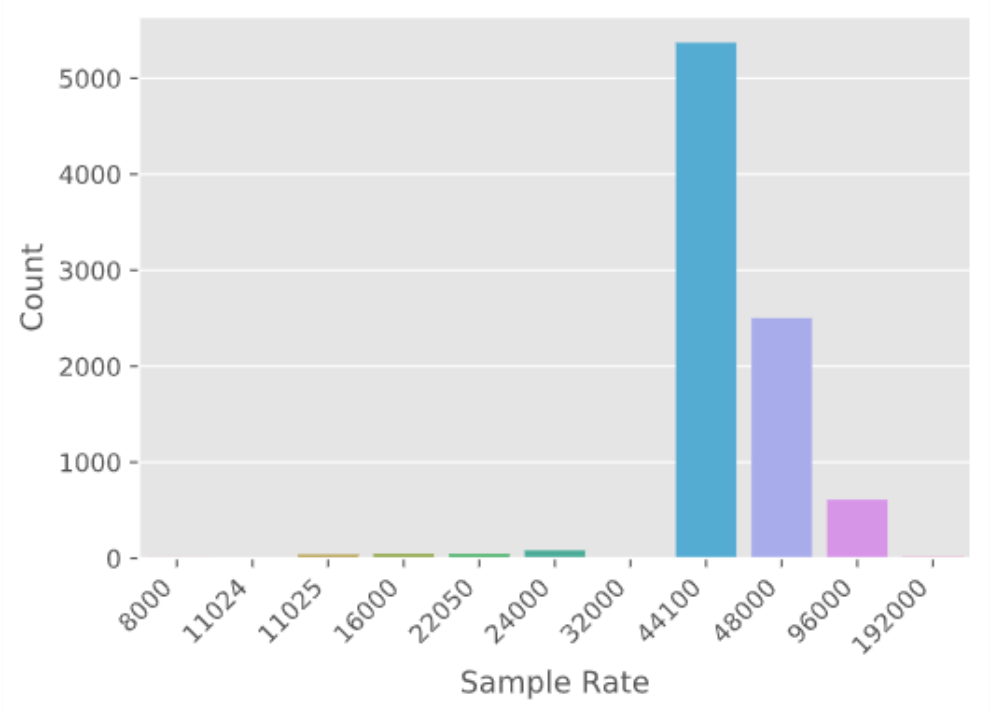


YORK UNIVERSITÉ UNIVERSITY

DATA PREPROCESSING (Contd')

- Varying bit depth in audio files was an issue.
  - Most libraries expect 16 bit files and we had to down/up sample to meet that requirement

- Huge variation in Sample rate across the classes. Needed to standardize to ensure features are comparable
  - The solution was to resample all files to 22,050 kHz sample rate



YORK UNIVERSITÉ UNIVERSITY

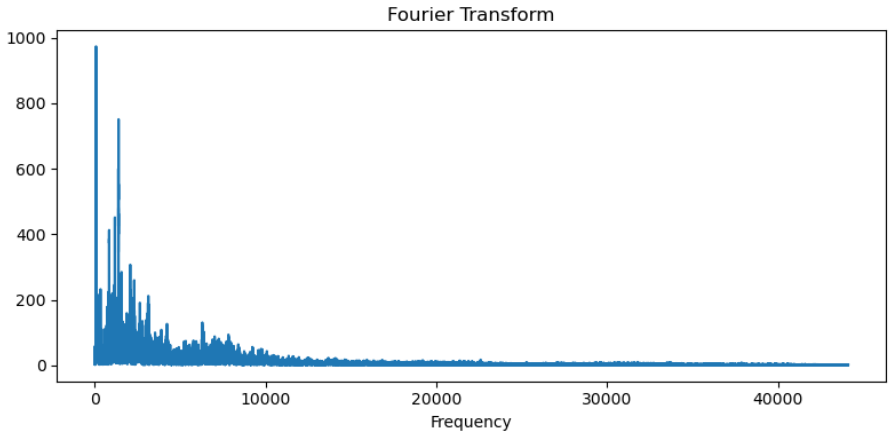
FEATURE ENGINEERING

We solved the problem in two different ways. One, to read raw audio files and use them in the models directly and another, to convert the audio into images and classify using the images. Consequently, the following features were engineered for further model building

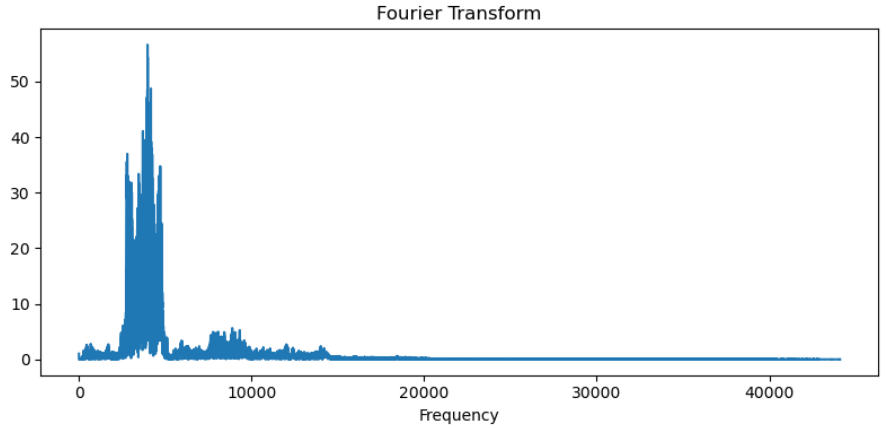
- MFCC
- Delta
- Mel Scale Spectrogram
- Chromagram
- Spectral contrast
- Short time Fourier transform
- Tonnetz
- Fast fourier transform (Images of plots of FFT)

FOURIER TRANSFORM

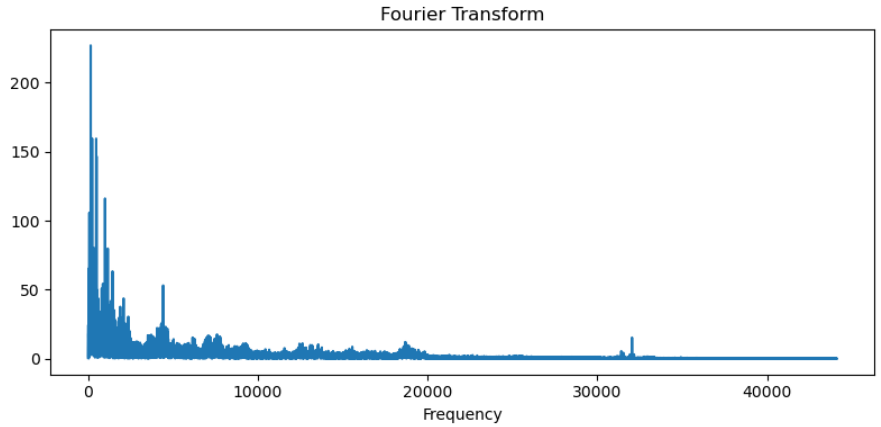
Fourier transform converts an audio file from time domain to frequency domain. The time domain doesn't offer much to analysis.



Air Conditioner



Siren



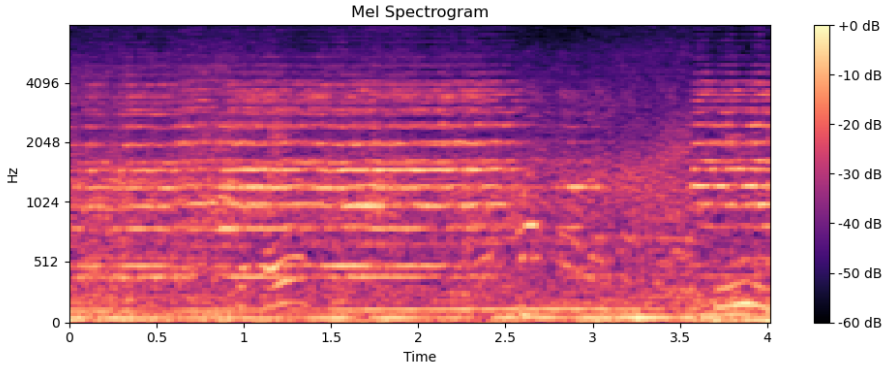
Children playing

SHORT TIME FOURIER TRANSFORM

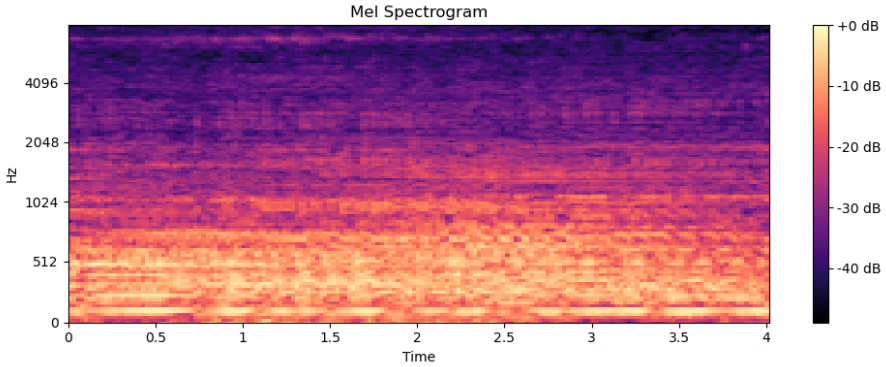
- A Fast Fourier Transform shows us the frequency distribution of a signal plotted against amplitude
- What if this varies over time? (Not just a solid tone)
- Solution: Short Time Fourier Transform
  - Combination of multiple FFTs taken over a multiple “windowed” segments of the file
- Visual representation of these combinations is a spectrogram

MEL SCALED SPECTROGRAM

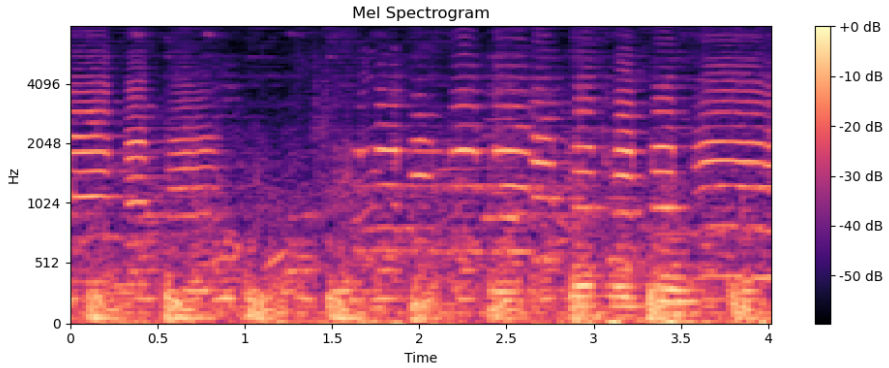
A spectrogram is a visual plot of frequencies in an audio sample as they vary over time. A Mel Scaled Spectrogram has had its frequencies converted to the Mel scale.



Car horn



Drilling

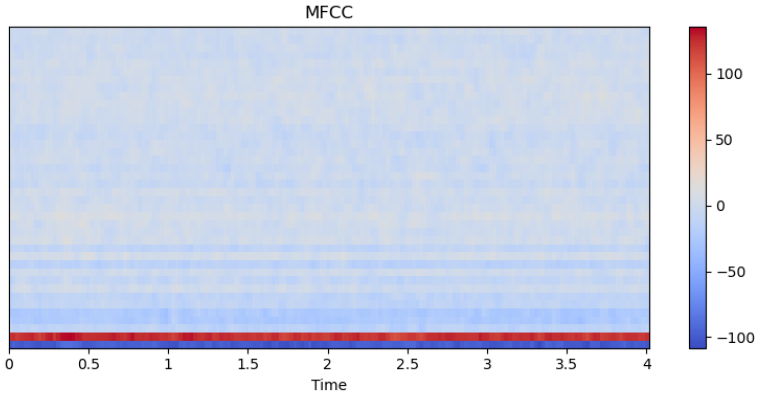


Street music

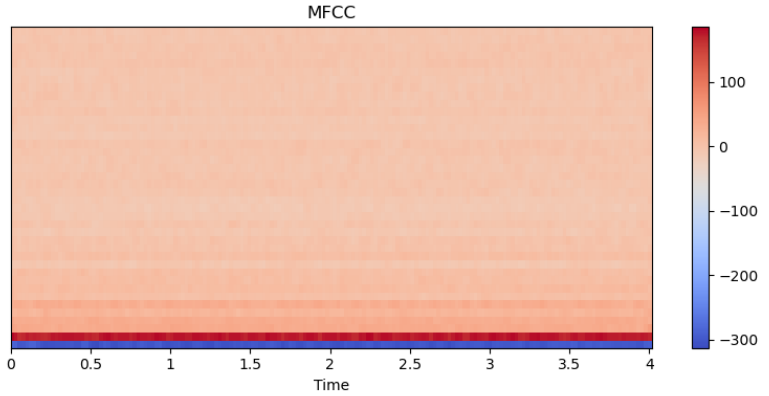
YORK UNIVERSITÉ UNIVERSITY

MFCC

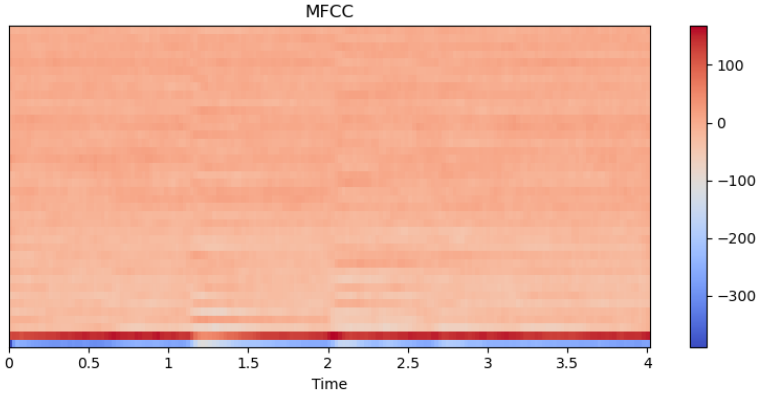
MFCC stands for Mel Frequency Cepstrum Co-efficients. The term 'cepstrum' is 'spectrum' in reverse. 'Cepstrum' is defined as the rate of change in spectral bands. Many-step process to generate: 1. split into frames, FFT, filter bank (Mel scale), logarithmic scale, FFT > set of cepstral coefficients that represent the audio signal. Idea is to represent specific, unique sounds in the MFC coefficients.



Air Conditioner



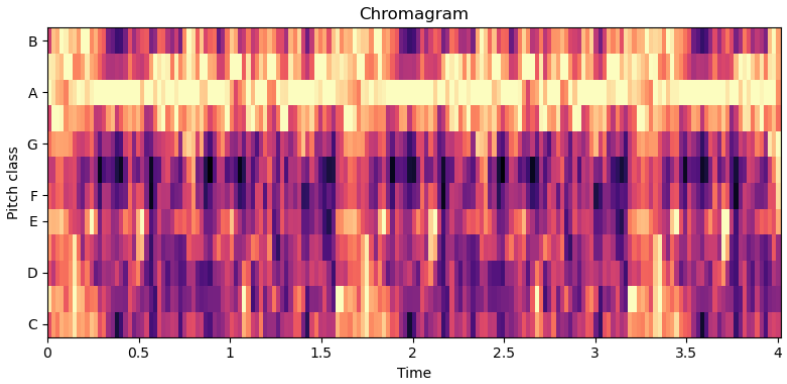
Engine Idling



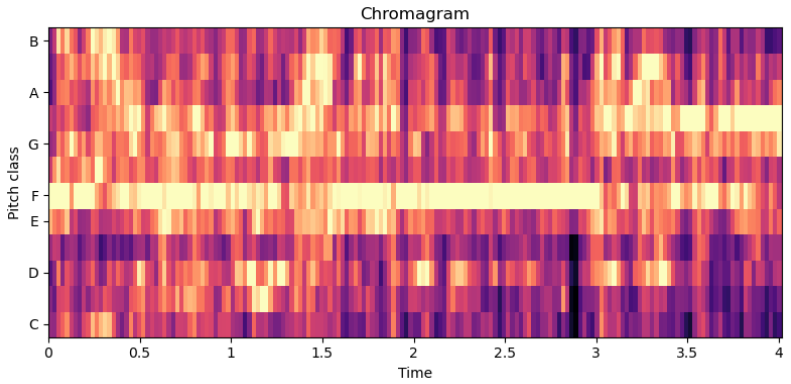
Dog Bark

CHROMAGRAM

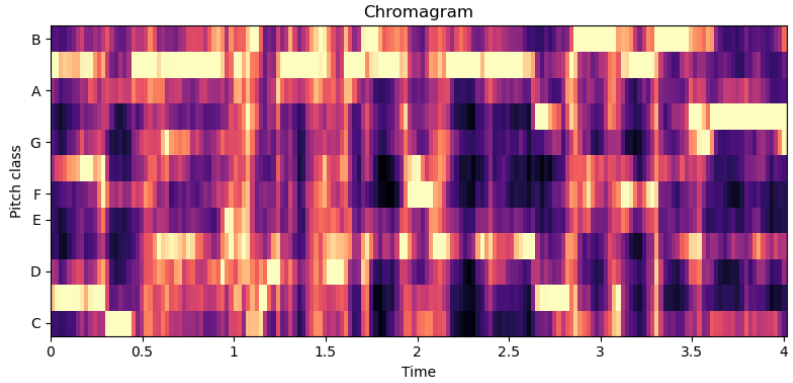
Chroma features are an interesting and powerful representation for music audio in which the entire spectrum is projected onto 12 bins representing the 12 distinct semitones (or chroma) of the musical octave.



Gun shot



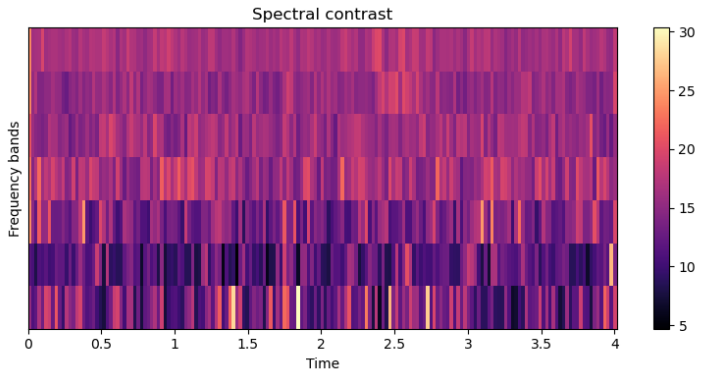
Jackhammer



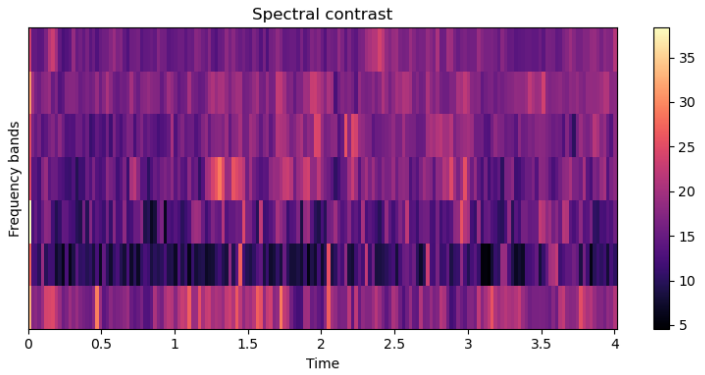
Street music

SPECTRAL CONTRAST

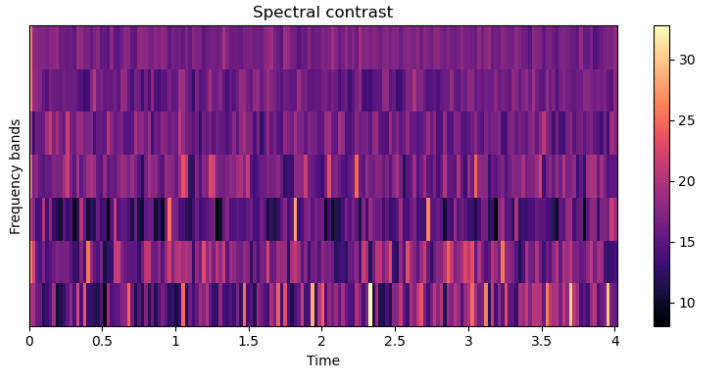
Input audio signal is split into multiple frames. For each frame, FFT is performed to get the spectral components and then it is divided into six octave-based sub-bands. Finally, Spectral Contrast is estimated from each octave sub-band. The raw Spectral Contrast feature estimates the strength of spectral peaks, valleys and their difference in each sub-band



Air conditioner



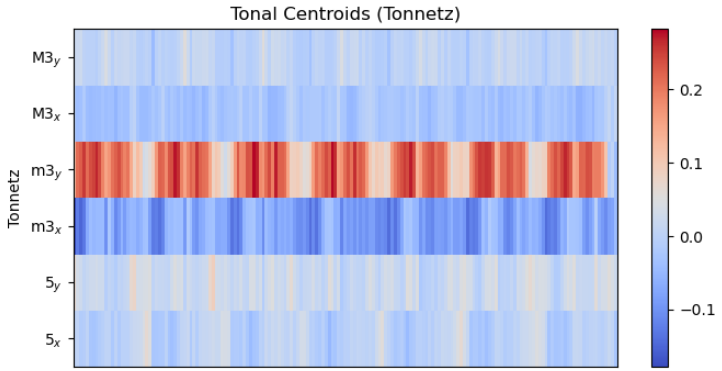
Children Playing



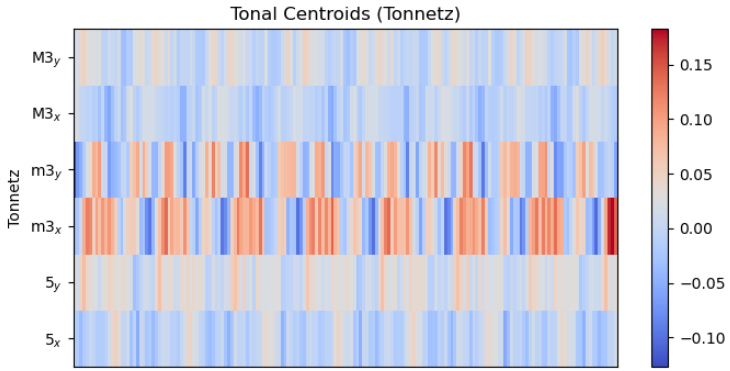
Engine Idling

TONNETZ

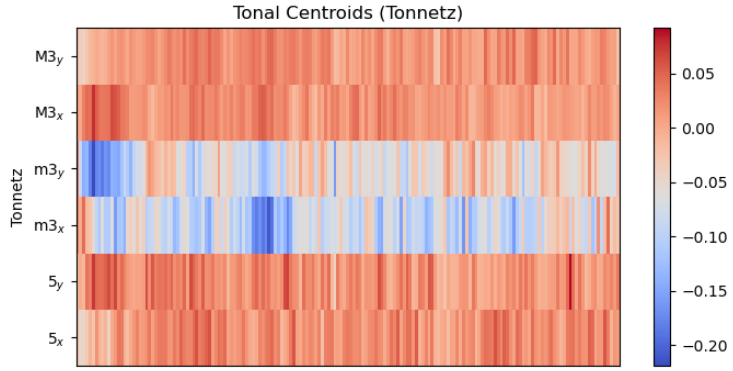
The Harmonic Network or Tonnetz is a well-known planar representation of pitch relations. For the purpose of audio classification, Tonnetz is used to compute tonal centroids as a feature. The six dimensional tonal centroid vector, v, for time frame n is given by the multiplication of the chroma vector, c, and a transformation matrix



Car horn



Dog Bark



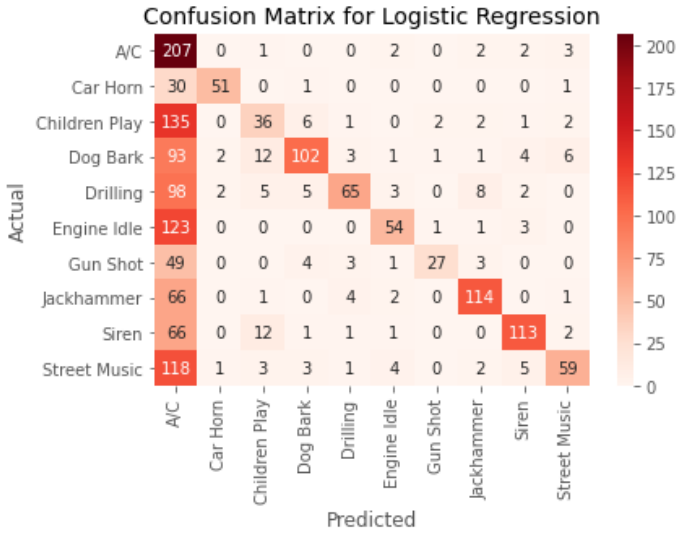
Jackhammer

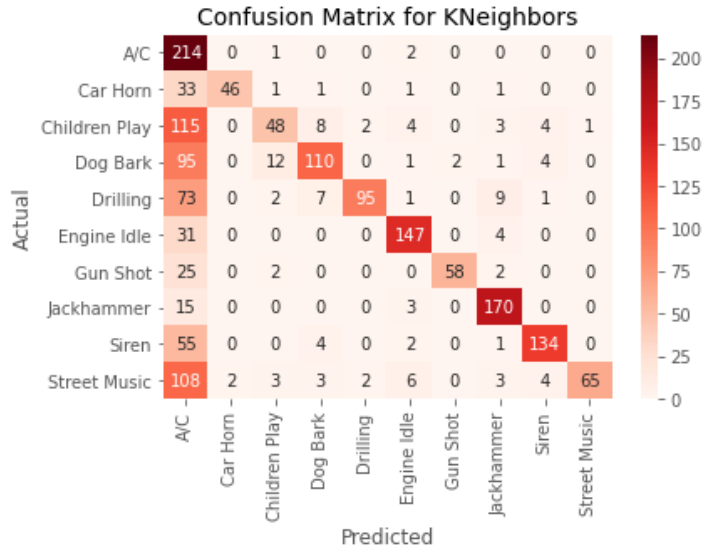
MODELS

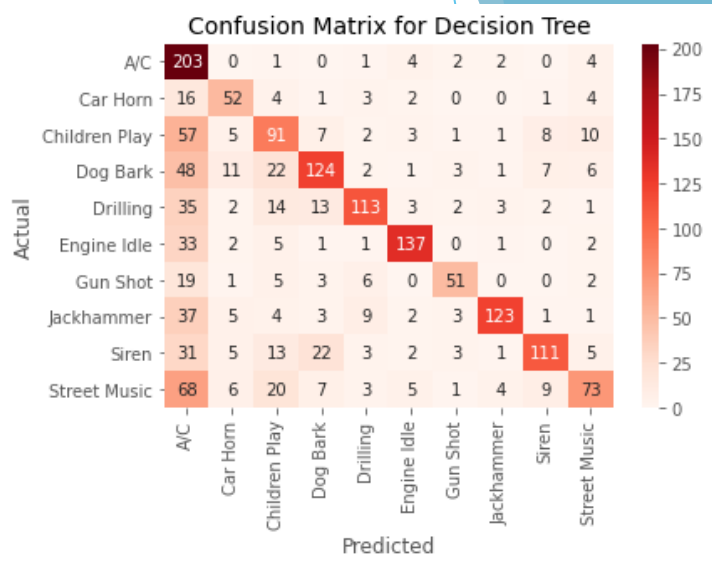
The following base models were run

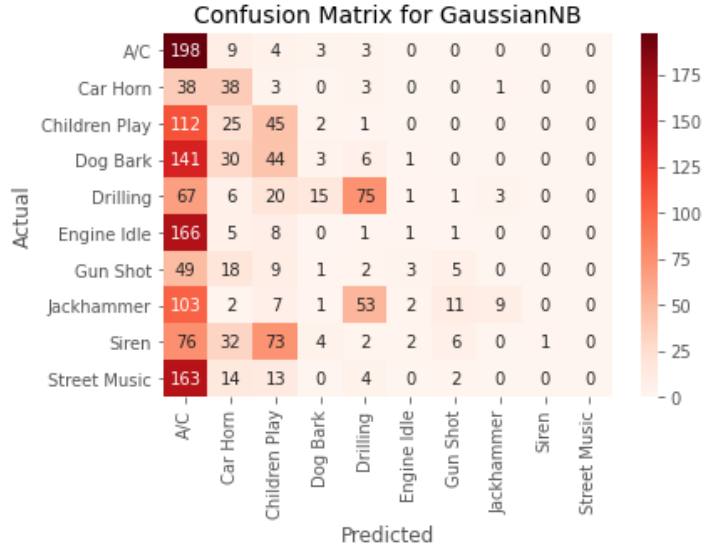
- Logistic Regression
- K Neighbours, K=10
- Decision trees
- Gaussian Naïve Bayes
- Linear SVM
- Bagging, Base learner = 100 Decision trees

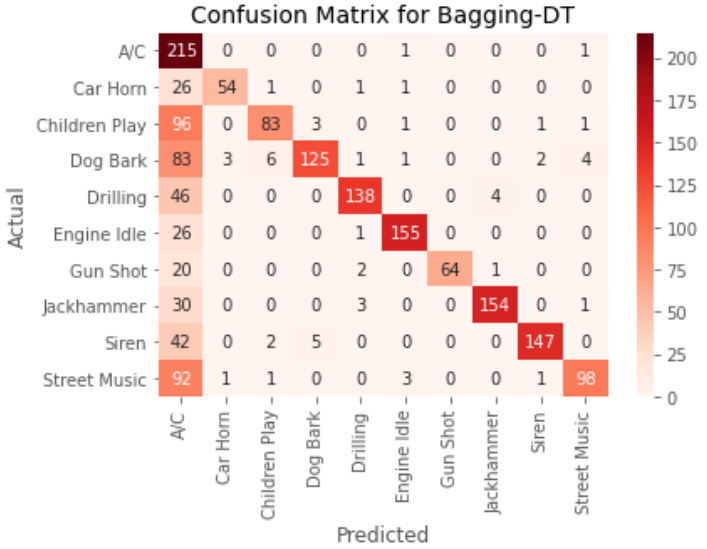
MODEL PERFORMANCE





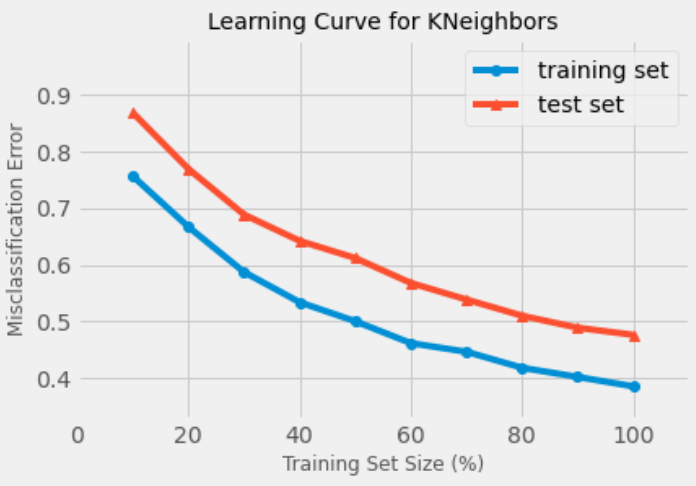


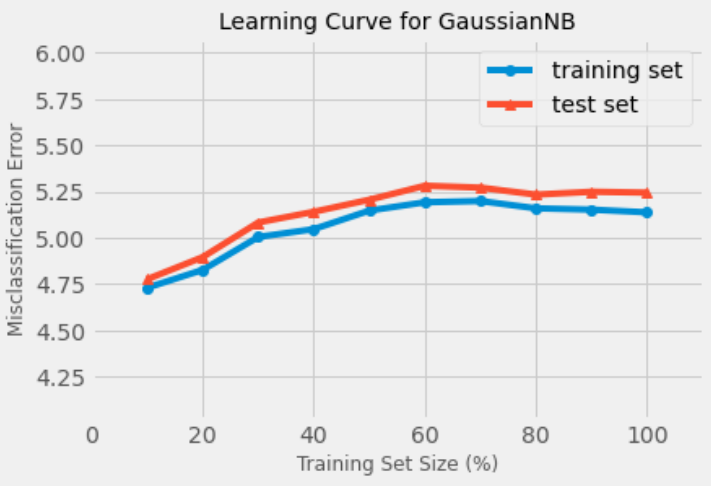


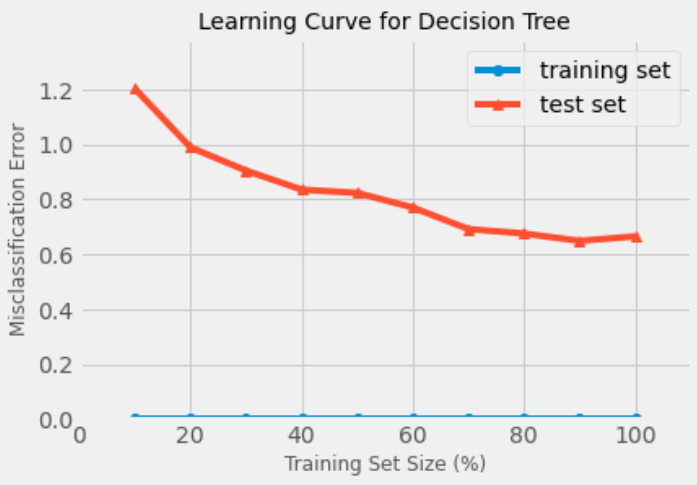


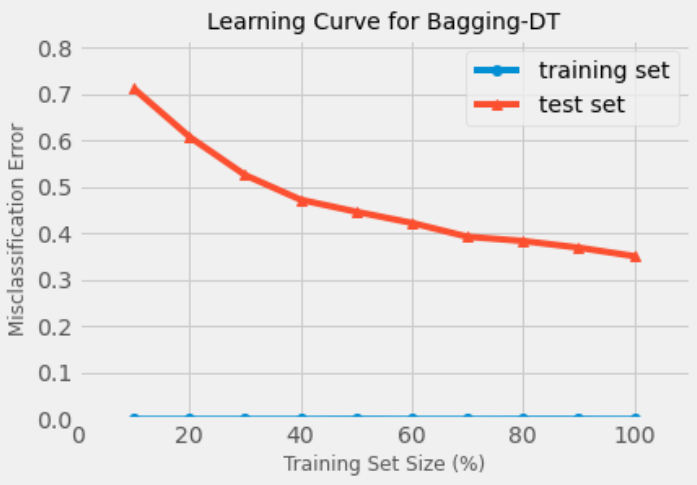
YORK UNIVERSITÉ UNIVERSITY

MODEL PERFORMANCE (Contd')



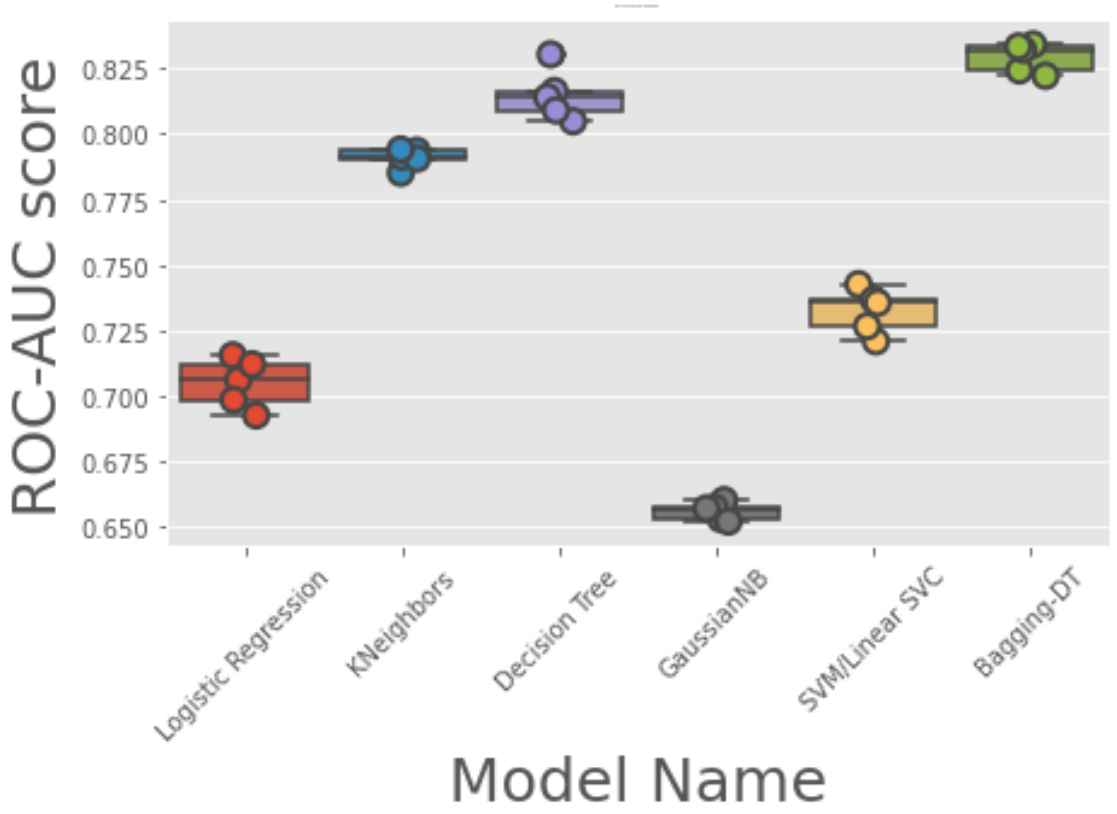


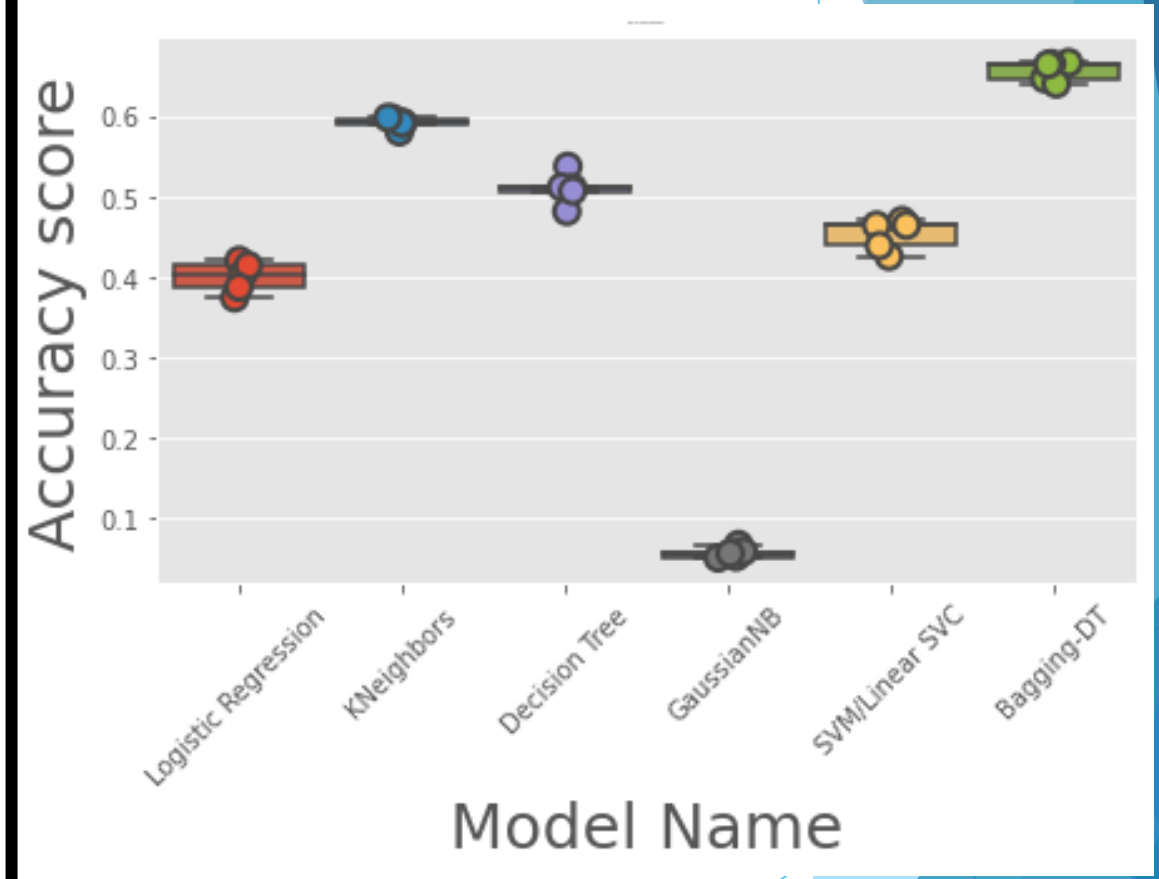




YORK UNIVERSITÉ UNIVERSITY

MODEL PERFORMANCE (Cont'd)





YORK UNIVERSITÉ UNIVERSITY

NEURAL NETWORKS

- We used 2 Neural Networks:
  - We built and trained a neural network from scratch to classify sounds with MFCC and its Delta as its features. The Delta we chose was the first derivative of the MFCC.
  - For Image Classification, we used a pretrained model called Inception to classify the FFT images of the sound dataset

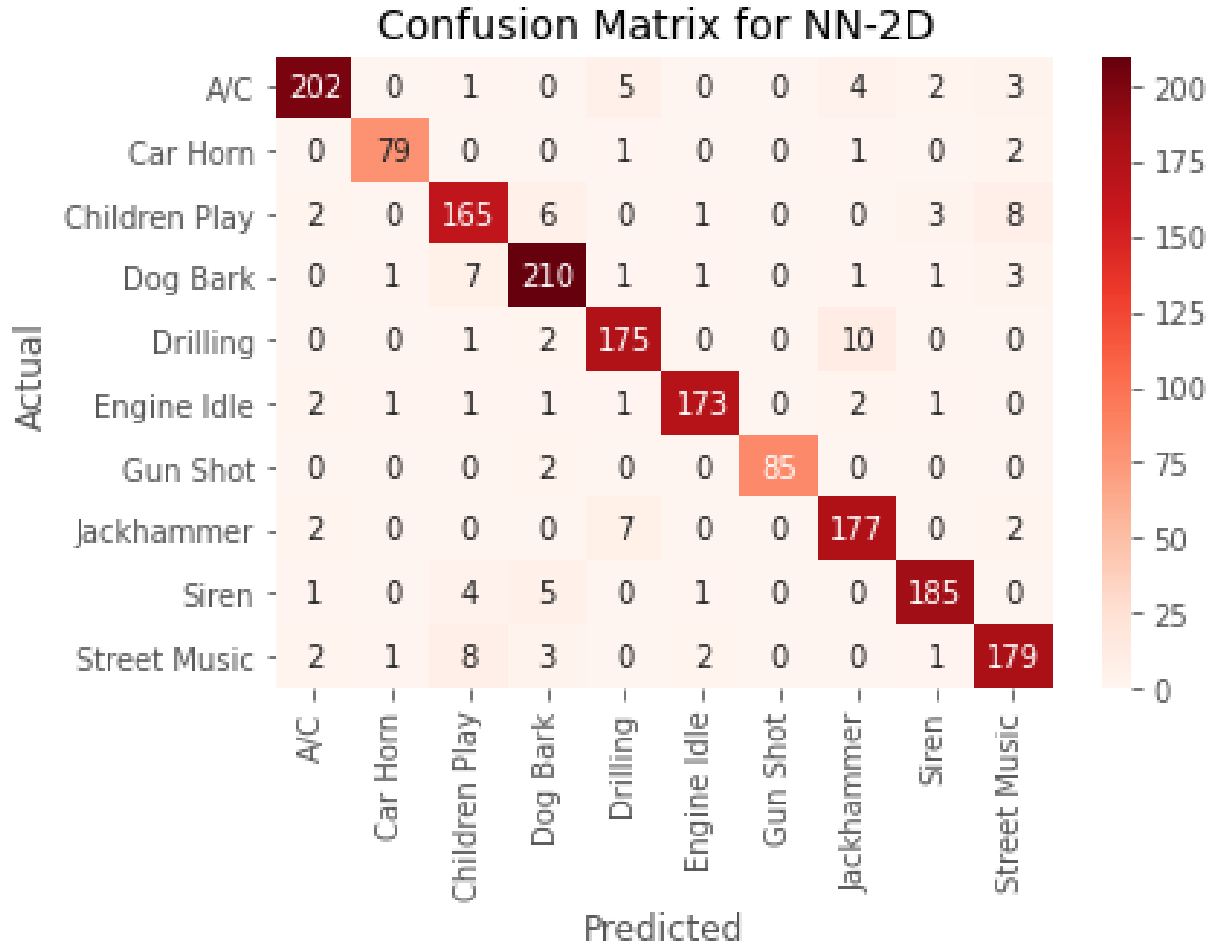
NEURAL NETWORKS – MFCC + Δ

- The combined "MFCC + Δ" feature has a shape of (40, 173, 2). This feature was placed into a very general 5 layer CNN. The results were astounding at 92%. Only in recent years have results of this quality been possible.
  - 40 - for the number of MFCC coefficients that are kept. This is a parameter.
  - 173 - the resultant data vector for each MFCC coefficient
  - 2 - MFCC in one channel, Delta in another channel

- The activation function we used was RELU.

- We used the ADAM optimizer, with a static 0.0001 learning rate.

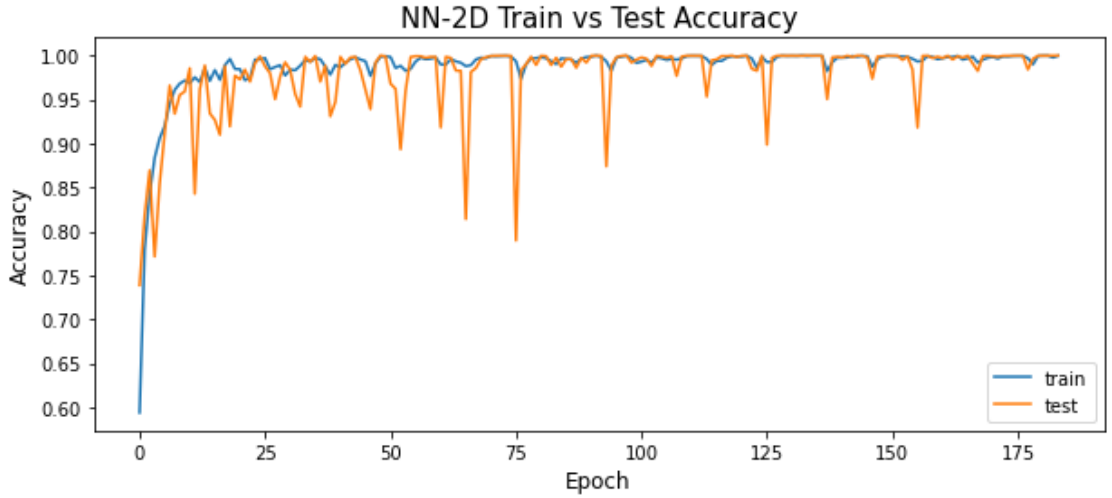
NEURAL NETWORK MODEL PERFORMANCE –
Confusion Matrix (MFCC + Δ)

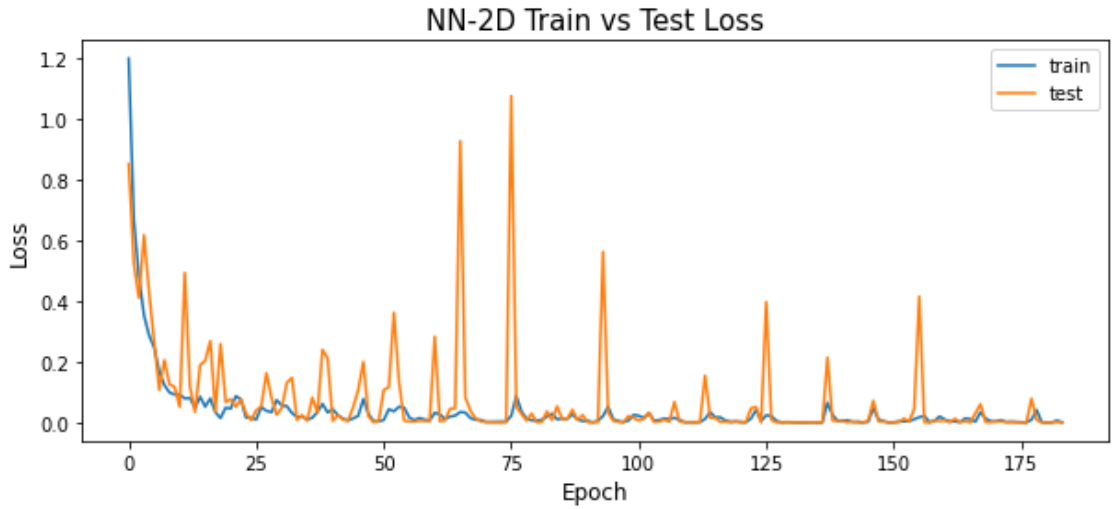


YORK UNIVERSITÉ UNIVERSITY

NEURAL NETWORK MODEL PERFORMANCE –

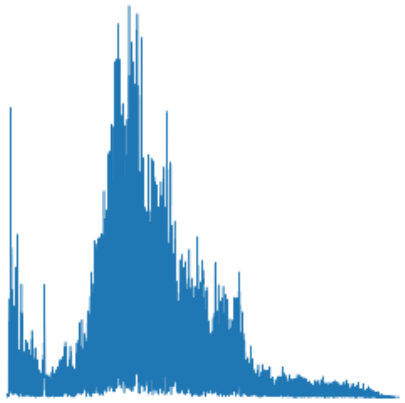
Accuracy & Loss (MFCC + Δ)



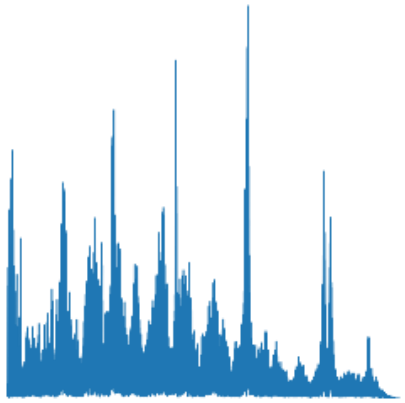


NEURAL NETWORKS – FFT Images, Inception V3

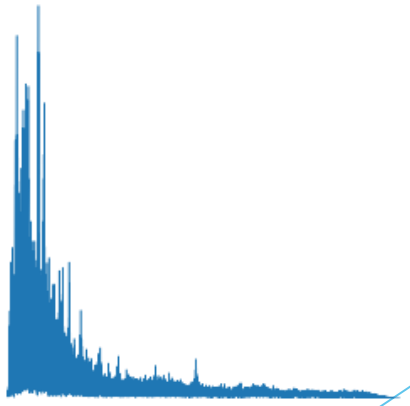
- All 8732 audio files were recursively read and put through an FFT. The FFT'd audio signals were plotted and saved as 256 x 256 images
- The dataset was split into train, test & validation sets, with 6286 samples for training, 1572 samples for test and 874 samples for validation or blind test
- While plotting, the axes and labels were removed as they're not useful information in classifying the images. Seen below are some sample FFT images
- This dataset was classified using the Inception V3 model which is pretrained on the ImageNet dataset



Dog bark



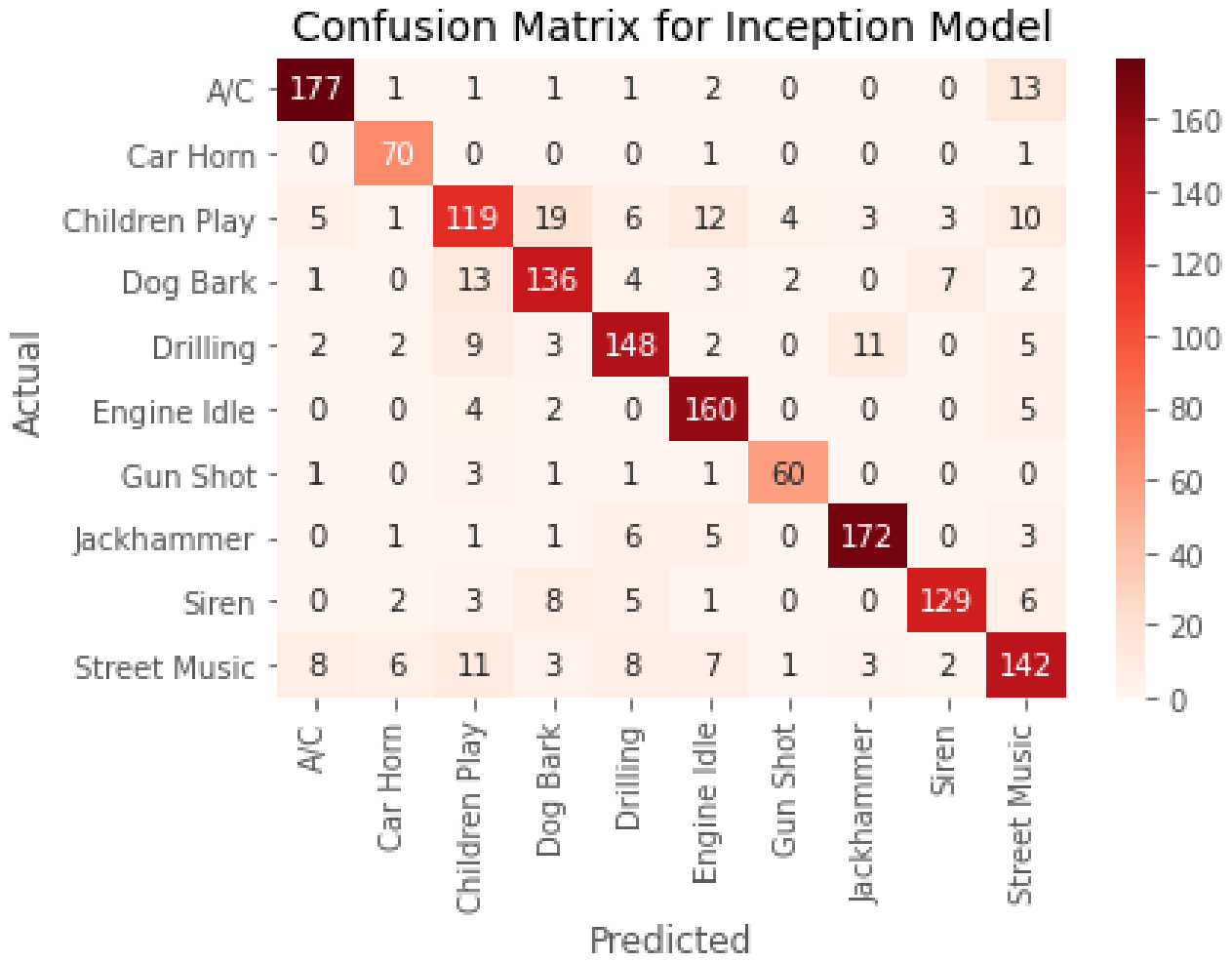
Jack hammer



Street music

NEURAL NETWORK MODEL PERFORMANCE –

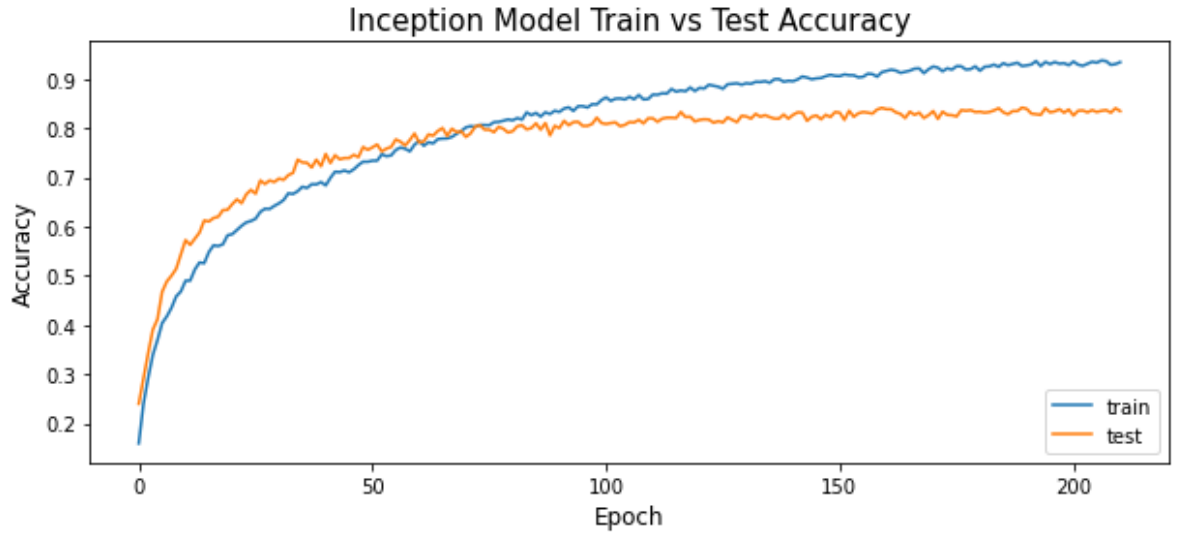
Confusion Matrix (FFT Images – Inception V3)

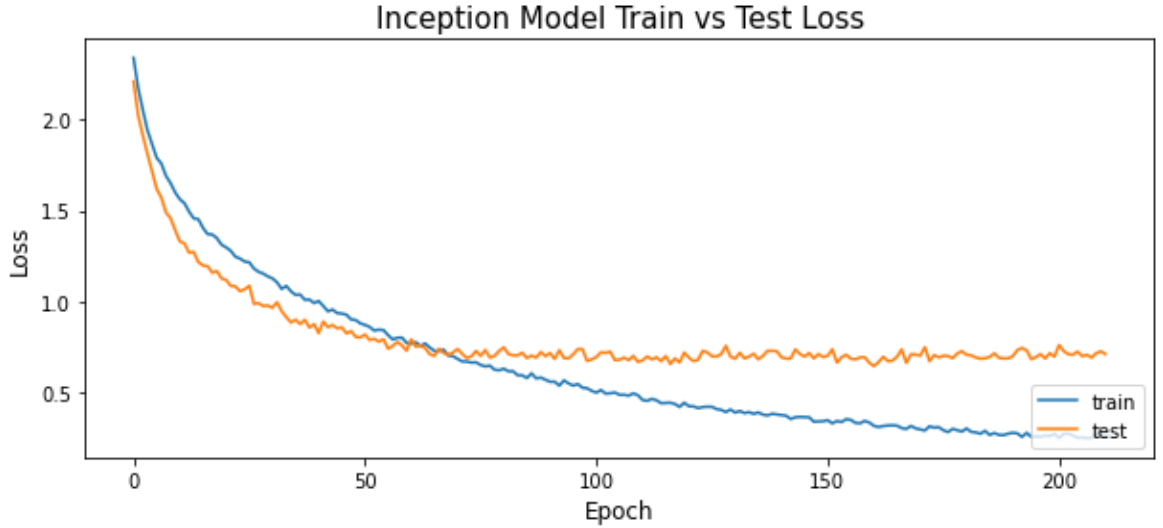


YORK UNIVERSITÉ UNIVERSITY

NEURAL NETWORK MODEL PERFORMANCE –

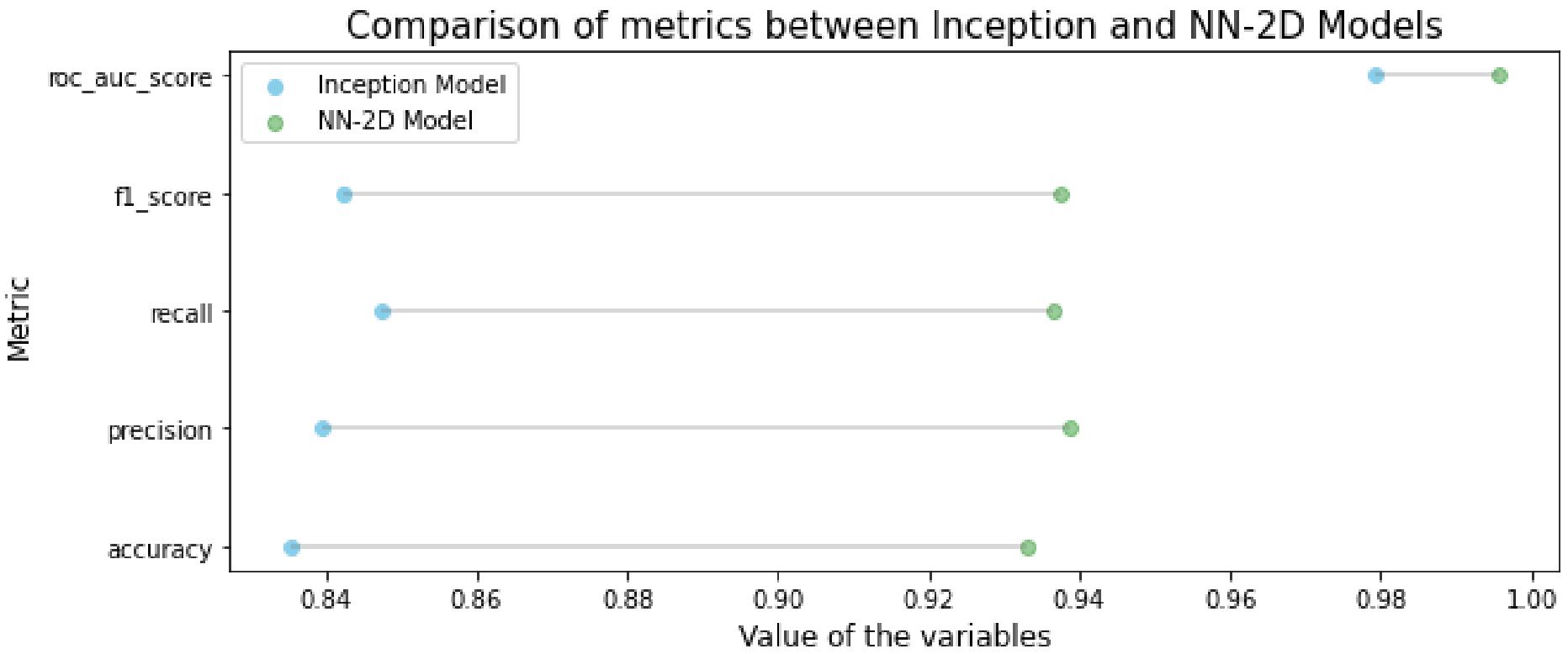
Accuracy & Loss (FFT Images – Inception V3)





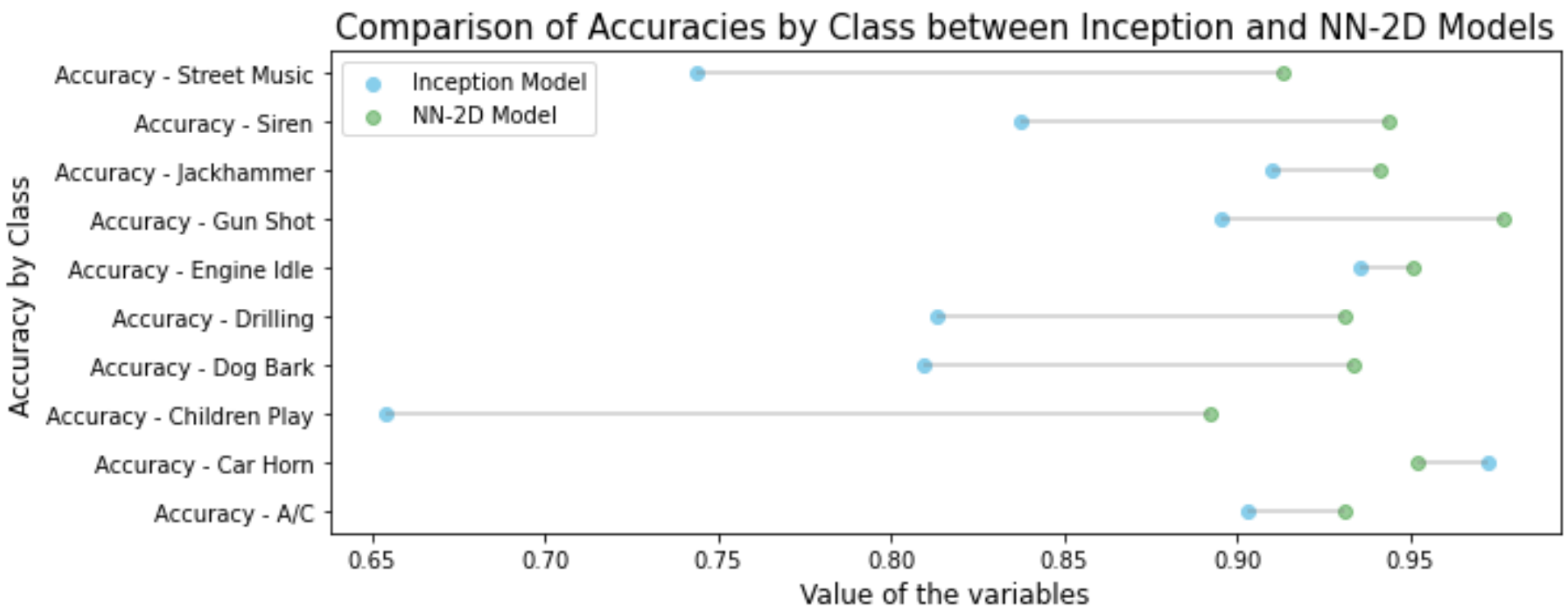
YORK UNIVERSITÉ UNIVERSITY

NEURAL NETWORK MODEL PERFORMANCE - Comparison of Metrics and Accuracies by Class



YORK UNIVERSITÉ UNIVERSITY

NEURAL NETWORK MODEL PERFORMANCE - Comparison of Metrics and Accuracies by Class



NEURAL NETWORK MODEL PERFORMANCE - Summary

- Both models performed well. However, the MFCC+Δ has better accuracy
  - Inception model: ~83 % accuracy
  - NN-2D model: ~93% accuracy
- Both models classified majority of classes well, with the exception of 'Children Playing' and 'Street Music'
- Learning Curves
  - Inception model has small overfitting issues
  - NN-2D model may need better hyperparameter tuning (e.g. learning rate)

CHALLENGES

- As sound classification is **nowhere near as common** as image recognition, we had to work out our approach from the ground up
  - Group not as intuitively familiar with audio processing
- Generating **numerical features**
  - Attempted to do this manually at first, creating waveforms with various packages, reading audio as array. N-Dimensional features were an issue
- Very **inconsistent audio files,** submitted by various users on FreeSound.org. Standardized collection?
  - Bit Depth, Sample Rate, length, salience (background/foreground)
- **Lack of pre-built features** to do exactly what we want
  - Built our own library for commonly used functions ("sonicboom.py)
- How to spin up **effective cloud resources** quickly
  - Used Google "Deep Learning VM" image and Deployment Manager + Jupyter Lab
- **Interpretability** of Neural Networks
  - Validation score issues in keras, solved by writing our own blind test function w/ holdout set

ETHICAL & PRIVACY CONCERNS

- Accidental capture/recognition of sensitive audio (private chats, personal info, unintended sounds, etc.)
- Acoustic fingerprinting (identifying location, other useful info from a specific set of recognized sounds, etc.)
- Code could be easily modified to classify and recognize speech but has not been designed with these considerations in mind
- Must be clear on the purpose for our code and that it is not to be used for purposes for which it is not intended

CONCLUDING SUMMARY

- From the metrics of model performance it is evident that the Neural networks performed far better than the conventional models

- If the solution were to be deployed, and if the requirement is very high accuracy, then we would choose the 5 layer CNN which gave an accuracy of ~93%

- If the requirement is high accuracy and also high interpretability, then we would choose the image classifier inception V3 model as the pretrained model has many existing work to leverage off of, including ways to plot & interpret the steps the NN takes the input through

FUTURE WORK

- What more do we want to do:
  - Further feature extraction, which is a rabbit hole for simple classification tasks, but is an even deeper hole for audio classification because the relationship of time and frequency is so complicated
  - Further Hyperparameter tuning of Models
  - Experiment with model architecture similar to newer, more complicated, cutting edge models
    - ex. Attention based DNN & Two-Stream CNN Based on Decision-Level Fusion (97% accuracy)
  - Investigate discrepancies in Class Accuracies, particularly classes 'Children Play' and 'Street Music' (the performance on these two are a bit lower for NN-2D and quite a bit lower with Inception)
  - Investigate "Salience" the difference in classifying audio samples that are in the background vs the foreground
  - Image Classification specifically: Ensemble methods + adding more features (multiple image features with ensemble voting)

THANK YOU

See report for references.