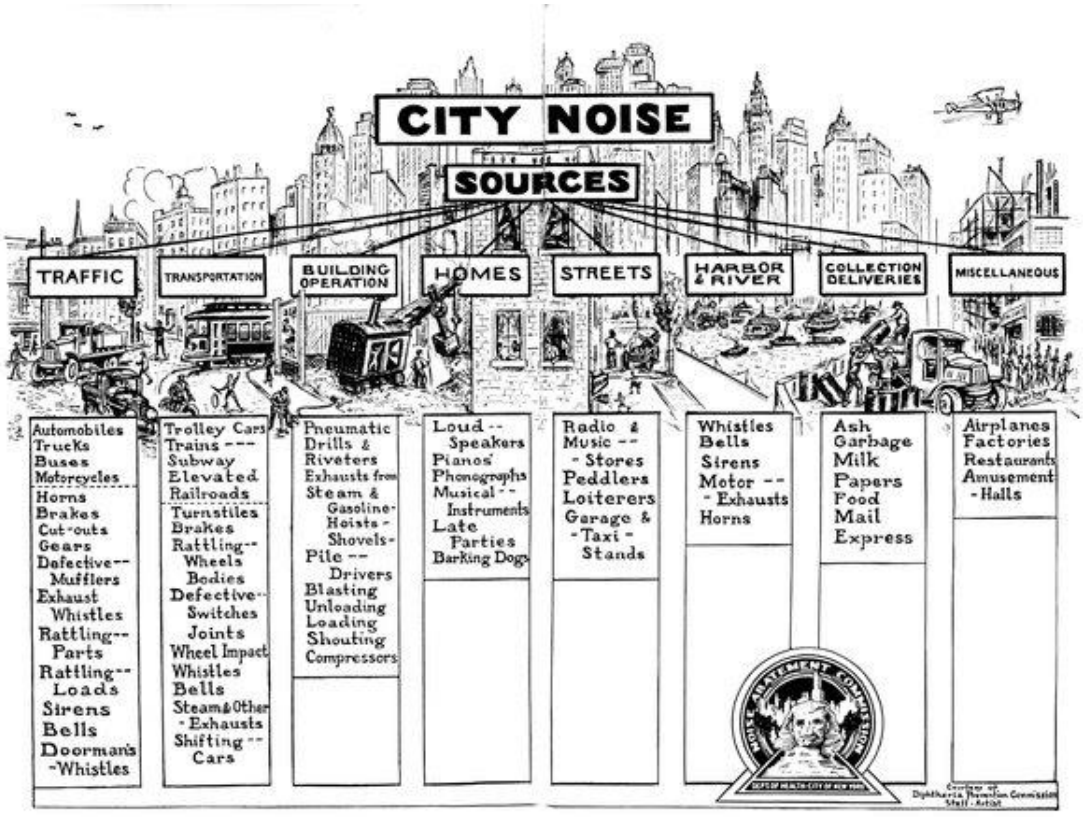
Urban Sound Classification
Final Capstone Project



Source: [Listening to the Roar of 1920s New York]

Abstract

We are in a world of sonic doom where noise pollution is one of the topmost quality of life issues for urban residents in the United States. It has been estimated that 9 out of 10 adults in New York City are exposed to excessive noise levels, i.e. beyond the limit of what the [EPA] considers to be harmful.

The objectives of this project is to evaluate and train various machine learning models to classify the urban sounds into categories correctly.

Introduction

Sonic event classification is a field of growing research. Most of these researches focuses on music or speech recognition. There are scarce works on environment sounds with very few databases for labeled environment audio data.

Source

Audio data for this project is collected from UrbanSound8k, released by NYU CUSP. The data was sourced from field recordings uploaded to the Free Sound online archive.

The sources were selected from the Urban Sound Taxonomy based on the high frequency with which they appear in noise complaints as determined from the data provided by New York City's 311 service (over 370,000 complaints from 2010 to date)

Since these are real field-recordings, it is possible (and often the case) for there to be other sources present in a slice in addition to the labeled source. All slices have been manually annotated with the source ID and a subjective judgement of whether the source is in the foreground or background.

To facilitate comparable research, the slices in UrbanSound8K come pre-sorted into 10 folds using a stratified approach which ensures that slices from the same recording will not be used both for training and testing, which could potentially lead to artificially high results.

Download Link: Urban Sound 8K Audio Dataset

Data Set Information

The dataset is comprised of 8732 slices (audio excerpts) of up to 4 s in duration extracted from field recordings crawled from the Free Sound online archive. Each slice contains one of 10 possible sound sources: **air conditioner, car horn, children playing, dog bark, drilling, engine idling, gun shot, jackhammer, siren, street music**.

The files are pre-sorted into ten folds (folders named fold1-fold10) to help in the reproduction of and comparison with the automatic classification results reported in the article above.
In addition to the sound excerpts, a CSV file containing metadata about each excerpt is also provided.

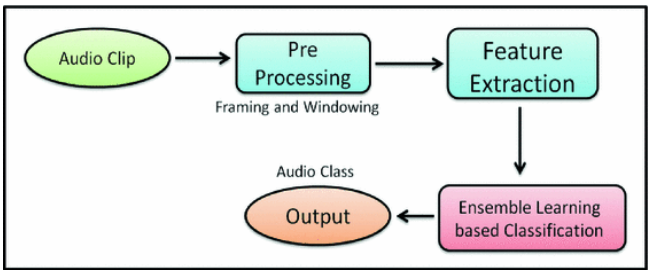
Problem Statement

Classify the audio files in urban setting and measure the performances of various models.

a. What feature extraction techniques should be used for optimal results?
b. How do the machine learning models compare against the neural network learning models?
c. Which model performed the best?

Approach

Fig A. Audio File Process



Source: Audio Files Process

a. First, perform exploratory data analysis on the audio files to quickly assess audio patterns.
b. Use feature extraction techniques for audio feature generation and embedding post processing.
c. Apply various machine learning based classification techniques to train the model to classify the audio file.
d. Evaluate and choose the best performer by measuring the effectiveness of different models.

Feature Extraction

Feature extraction is the most important part for designing a machine learning model.

To extract the useful features from sound data, we used Librosa and VGG Audioset library. These libraries provides several methods to extract different features from the sound clips. We explored the below mentioned methods to extract various features:

1. Mel-frequency cepstral coefficients (MFCC)
   - This method is available in Librosa Library.
   - It can extract 20-dimensional features from an audio file.
   - And is widely used in automatic speech and speaker recognition.

2. Visual Geometry Group (VGG, also Known as VGGish)
   - This method is available in the Audioset Library.
   - It can extract 128-dimensional features from an audio file.
   - A pre-trained convolutional neural network.

These two methods were compared in the previous clustering project "Unsupervised Learning".

In this capstone, we decided to use the VGG-like model available in the Audioset library available in the TensorFlow models Github repository, along with supporting code for audio feature generation, embedding post processing, and demonstrations of the model in inference and training modes.

VGGish was used as a feature extractor, it converts audio input features into a semantically meaningful, high-level 128-D embedding which can be fed as input to a downstream classification model. The downstream model can be shallower than usual because the VGGish embedding is more semantically compact than raw audio features.

Fig B. Distribution of Audio Classes

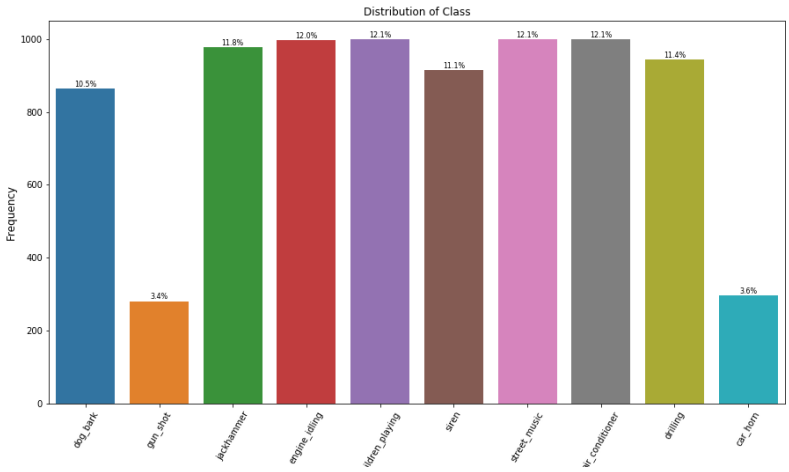
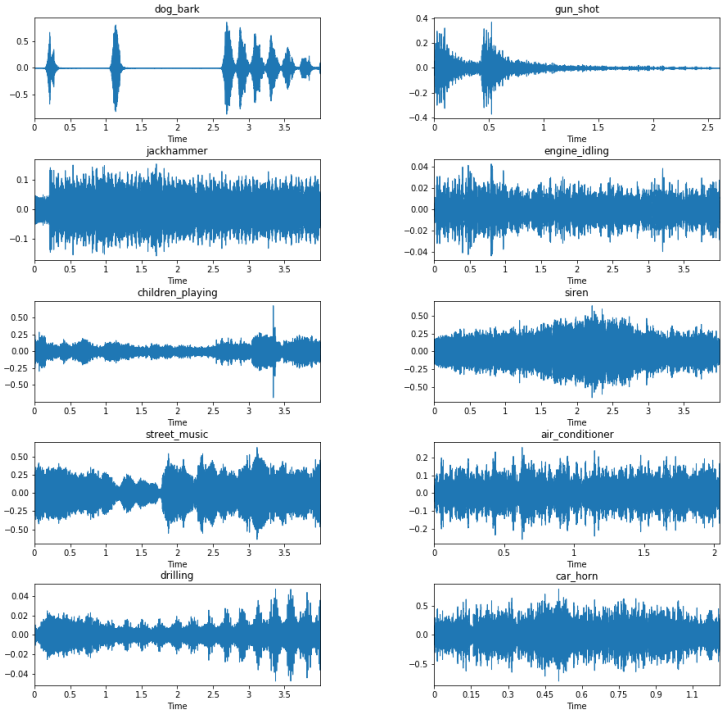


Fig C. Wave Plots



Conclusions

We trained 4 different models with hyperparameter optimization – Support Vector Machine, Random Forest, Deep Neural Networks and Convolutional Neural Networks.

Support Vector Machine model performed better than all others with training and test accuracy of 93%.

Fig D. Confusion Matrix

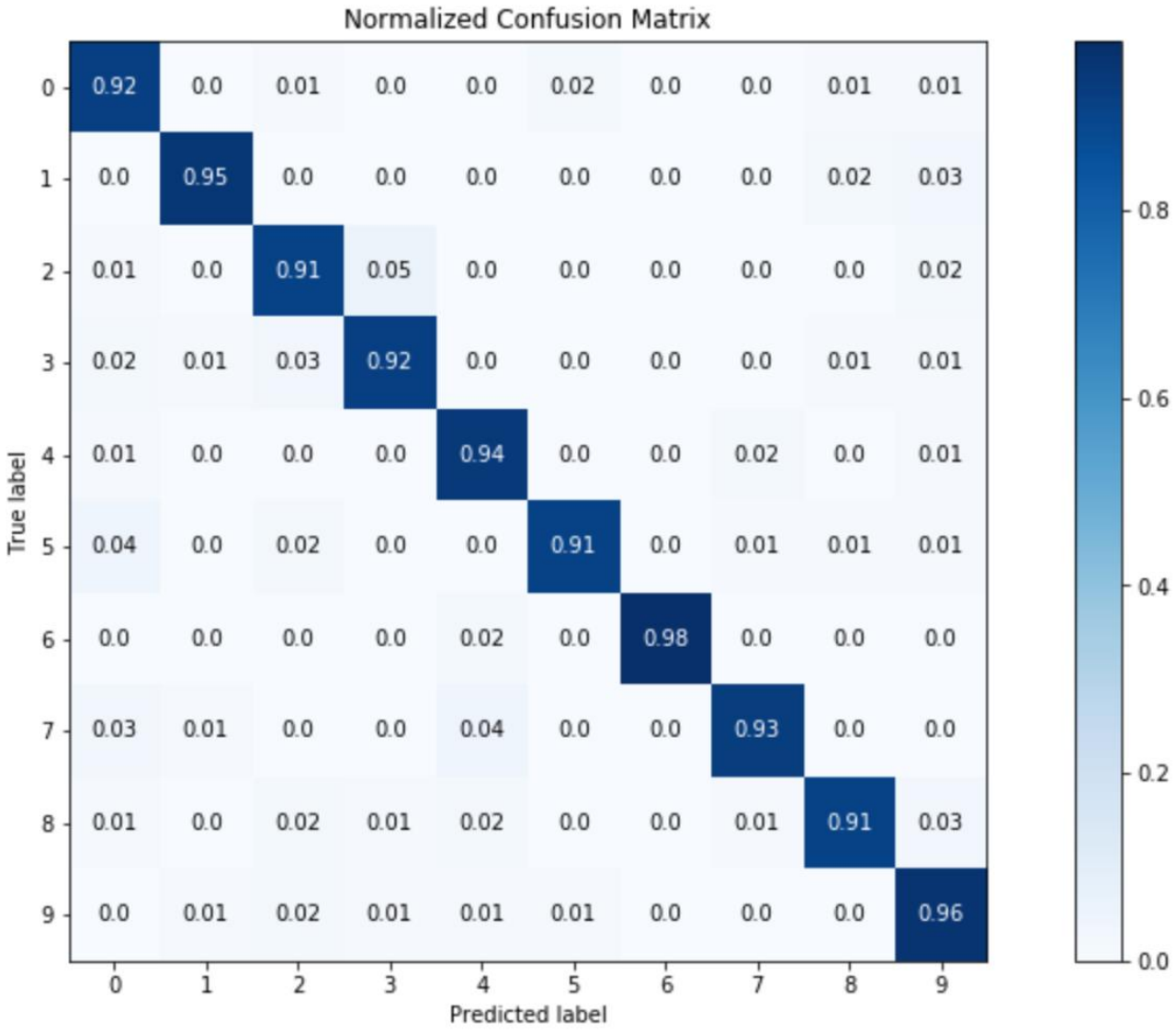
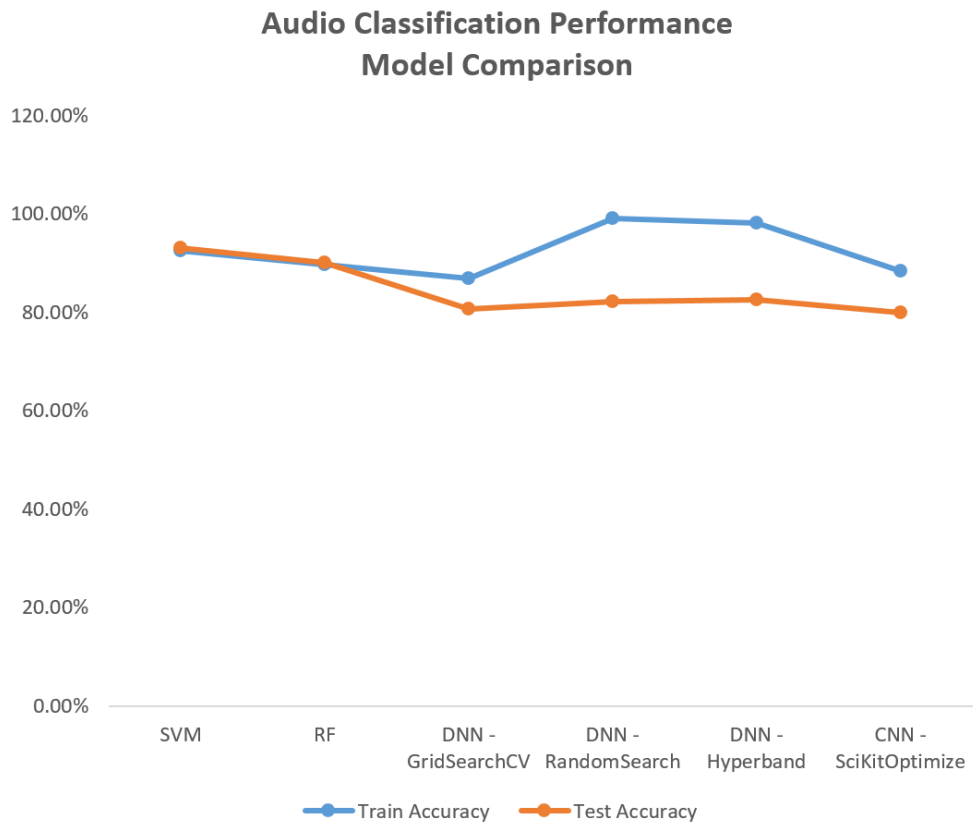


Table - Model Comparison

Model	Hyperparameter Optimization	Training Accuracy	Test Accuracy	Issues
Support Vector Machine	GridSearchCV	92.46%	93.00%	-
Random Forest	GridSearchCV	89.62%	90.00%	-
Deep Neural Network	GridSearchCV	86.90%	80.60%	Overfitting
Deep Neural Network	Keras Tuner - RandomSearch	99.00%	82.20%	Overfitting
Deep Neural Network	Keras Tuner - Hyperband	98.15%	82.50%	Overfitting
Convolutional Neural Network	Sci-Kit Optimize	88.36%	79.91%	Overfitting

Fig D. Model Comparison



Summary

- Support Vector Machine SVM model performed better than all others with training and test accuracy of 93% followed by the Random Forest model with 89% accuracy.
- Both DNN and CNN seems to have overfitting issues based on their test accuracy score lower than the training accuracy score.
- Having the lowest number of samples, gunshot is still managed to have the highest proportion for true positive value for all the models.
- Car horn and gunshot have less than 300 samples compare to other classes, which have around 1000 samples each

Practical Uses

The automatic classification of audio events in an urban setting has a wide range of applications in areas such as,

- Audio Event Detection
- Audio Music Tagging
- Audio Fingerprinting
- Music Retrieval
- Music Recommendation
- Home security or Audio Surveillance
- Assisted living, elder or infant care
- Accident and crime surveillance

For example, a baby crying, a person screaming, people arguing that may lead to violence, sound of a gunshot/explosion or someone calling for help are just some audio events that require action. Manually monitoring for these sounds, either in close proximity or remotely through a monitoring device, not only demands attention but also requires the person to be within hearing distance. This is not always possible, and is where audio event detection, or sound recognition, solves real problems. It will automatically alerts an application if a specific sound is detected, so that a human may take the appropriate action. Other applications of this project includes music classification, detecting sounds of different species for wildlife preservation etc. without engaging human resources to classify them.

Future Considerations

This capstone project focuses on the various machine learning techniques to model the data to give us predictive power to classify the sonic events accurately. Improving this model to optimize prediction of the audio classification includes supervised machine learning models such as Random Forest, and Support Vector Machines as well as neural network models such as Deep Neural Networks and Convolutional Neural Networks using TensorFlow and Keras.

By modeling and interpreting the data, we can potentially improve the quality of life of city dwellers by providing a data-driven understanding of urban sound and noise patterns, partly enabled by the move towards *"smart cities"* equipped with multimedia sensor networks.

References

1. VGGish Audio Embedding Colab
2. Hyperparameter Optimization in TensorFlow
3. Hyperparameter Tuning using Keras Tuner

Technologies Used

- Python, Numpy, Pandas, Matplotlib, Seaborn, Sci-Kit Learn, Librosa, Audioset
- Jupyter Notebook