Urban sound classification

Francesco Tomaselli

July 20, 2021

Contents

1 Introduction

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

The goal of this project is to build a neural network to classify audio files from the *UrbanSound8k* dataset.

This dataset contains audio divided in ten classes, each one representing a different type of city sound, for instance, we can find *car horns*, *dogs barking*, *sirens*, etc. A deeper discussion about the dataset is present in the Subsection 2.1 on the following page.

The presented methodology is composed of three main parts. The first step is to extract relevant features from audio files using the *Librosa* library. This is discussed on Section 2 on the next page.
The next step consists in composing and refining a neural network to classify the data obtained from the previous step. This part is made possible by the *Keras* library and it is discussed in Section 3 on page 4.
Lastly, results from the classification, namely accuracy and standard deviation among test sets, are presented in Section 4 on page 4.

The project is developed in *Python* and the code is structured in a *src* package. Each one of the sub-packages contains code to deal with the different parts of the project. For instance, the *data* folder contains the classes to extract features and to manage a dataset.

In addition to the package there is a *notebook* folder containing the different steps of the project and the various experiments made.

2 Feature extraction

This Sections presents the original dataset structure and the steps followed to create training and test sets from it.

2.1 Dataset structure

The dataset contains ten folds of audio samples, each one about four seconds long. The samples are divided in ten classes among ten folds.

The training set consists of the first four folds plus the sixth, the other folds create five different test sets.

The classes presents

2.2 First dataset

Choosing the features to extract was difficult as I did not have prior experience working with audio.

The *Librosa* library provides many feature to choose from, for my first try with this dataset I kept it simple by opting for these three ones:

1. *Mel-frequency cepstral coefficients*
2. *Chromagram*
3. *Root-mean-square*

Each feature consists of an array of arrays containing measurements. I applied a series of functions to each sub-array and then concatenated the results in a final feature vector. The functions applied are *minimum*, *maximum*, *mean* and *median*.

This approach resulted in 132 components feature vectors.

Feature scaling After testing some neural networks on the first dataset the results were not promising. One of the reasons is the big difference in ranges among feature vector components.

. . .

To mitigate this effect scaling was applied to the dataset. This lead to an improvement on the results using the same model as before.

The scaler trained on the training set was then used to scale the five different tests sets.

2.3 Extended dataset

To improve results on the test sets new features were added to the dataset, namely:

1. *Zero-crossing rate*
2. *Roll-off frequency*
3. *Spectral flux onset strength*

After applying the same four functions to these three new arrays, a total of 12 new features were added to the dataset. Also, as scaling yield to promising results, the same scaler was used on this new dataset.

After testing a really simple network on the new dataset results improved once again.

PCA Scaling lead to important improvements on the first dataset, so I decided to try the PCA to select the most important features from the extended one.

. . .

3 Model definition

3.1 Neural network structure

3.2 Hyperparameter tuning

4 Results

4.1 First dataset

4.2 Extended dataset

5 Final remarks