Implementation of SED with Depthwise Separable and Dilated Convolutions

Neural Networks Sapienza 2020

Riccardo PRINZIVALLE

March 2021

1 Introduction

This project is a study and implementation of a polyphonic sound event detection extracted from [1]. It is also based on the baseline reference of [1], which is [2]. These two works represent the main source of this project. Here there will be presented both a replication of the paper approach together with a monophonic sound event detection, since to obtain the original dataset took some time, the author thought to start working with another dataset and then move the work to the original dataset when it would have been available. Section 2 and 3 are organized as follows: first an analysis of the dataset is performed to better understand it, then it is explained how the feature have been extracted and finally it is proposed a model to solve the problem. Section 4 regroups the results for both datasets, then it is explained a brief digression on how to train a neural network model on an AMD GPU on section 5 since the author's setup has only an AMD GPU. The work is ended by conclusions of section 6.

2 Monophonic SED

Monophonic Sound Event Detection consist of predicting a single label for an audio recording: the record will likely contain some noise but it generally contains a single and remarkable sound to be identified. In this case, it is used the *UrbanSound8K* dataset [3].

2.1 Data analysis

The dataset is composed by 8732 labelled small sound recordings (less than 4 seconds) from 10 classes: *air_conditioner*, *car_horn*, *children_playing*, *dog_bark*, *drilling*, *engine_idling*, *gun_shot*, *jackhammer*, *siren*, and *street_music*. The classes are balanced except for some, it can be seen in table 1. Only 3 out of 10 classes have less than 1000 elements, so there can be some problems predicting these classes.

Label	number of elements
air_conditioner	1000
car_horn	429
children_playing	1000
dog_bark	1000
drilling	1000
engine_idling	1000
gun_shot	374
jackhammer	1000
siren	929
street_music	1000

Table 1: Monophonic dataset label distribution

Moreover, the recordings have different properties since the come from and are taken as they are. The first difference is in the audio lengths visible in figure 1: the majority of audio have a duration of about 3.5/4 seconds, but there exists also smaller recordings which are in a tiny number.
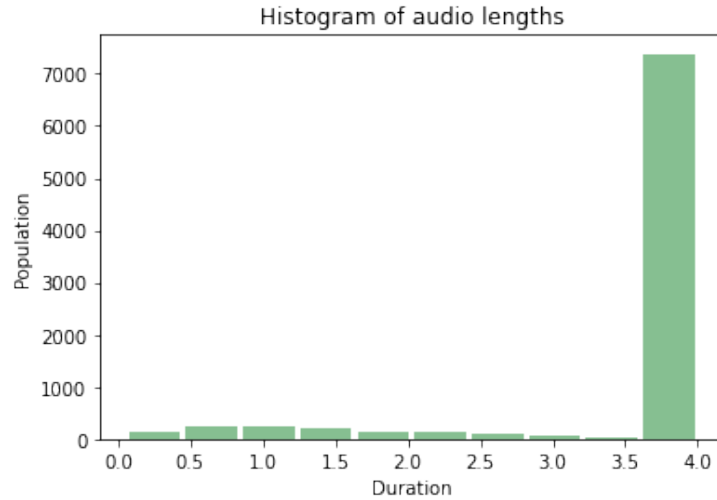

Figure 1: Audio duration distribution

The main differences are in bit depth, from 4 to 32 bit, the majority with 16 bit; and in the sample rate, from 8 KHz to 192 KHz, with the majority with 44.1 KHz. This may be a concern since some audio have a poor quality which can translate in poorer feature w.r.t. the other tracks. All these differences will be equalized during the feature extraction phase.

2.2 Feature Extraction

This phase adopts librosa [4], which is a sound processing library for python. Its use helps to deal with different audio characteristics since by default librosa converts audio to 22 KHz sampling rate and 16 bit depth. Since the majority of audio recordings are at 44.1 KHz, it may seem that down sampling may reduce audio quality, but if we visualize the sound with a spectrogram, it will be clear that most of frequency content is distributed well below the 11 KHz (which is the maximum frequency a 22 KHz sampling rate can process), so in this case it reduces the dimension of the data without losing much information. For what concerns the bit depth, the majority of recordings are already at 16 bit, so it does not change much the data. Audio file are loaded and transformed into array series by *load* function, which is also responsible of audio conversion and standardization.

The reference paper [1] uses Mel Frequency Cepstral Coefficients (MFCC) to extract features from the array sound data. MFCCs are a way of measuring the rate of information change in spectral bands and storing it in coefficients; moreover, the rate of change is modeled in a non linear way since the Mel band is logarithmic and the adoption of this band is able to capture the rate of change in a similar way to what the human hear does [5].

2.3   Model formulation
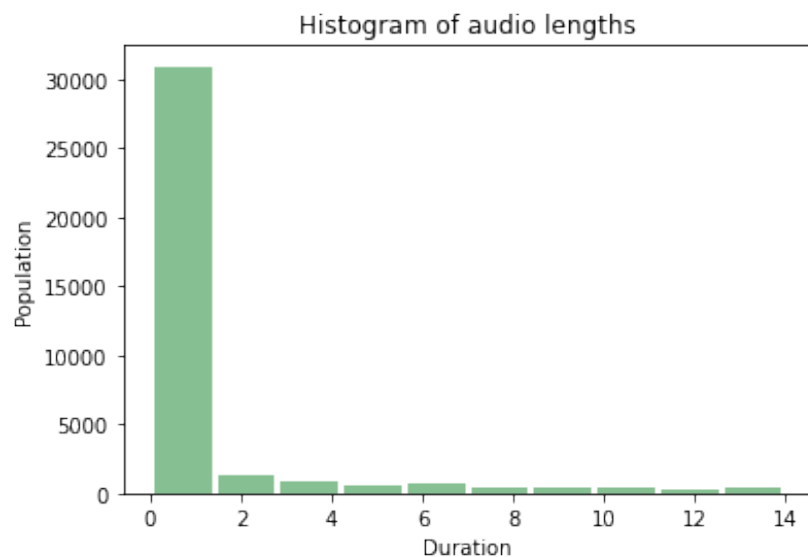
3   Polyphonic SED

3.1   Data analysis

Figure 2: Decision trees assembly only confusion matrix

3.2   Feature Extraction

3.3   Model formulation

4   Experimental Results

4.1   Monophonic results

4.2   Polyphonic results

5   How to train NN on AMD GPU

6   Conclusions

References

[1] Konstantinos Drossos, Stylianos I. Mimilakis, Shayan Gharib, Yanxiong Li, and Tuomas Virtanen. Sound event detection with depthwise separable and

dilated convolutions, 2020.

[2] Emre Cakir, Giambattista Parascandolo, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen. Convolutional recurrent neural networks for polyphonic sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(6):1291–1303, Jun 2017.

[3] J. Salamon, C. Jacoby, and J. P. Bello. A dataset and taxonomy for urban sound research. In *22nd ACM International Conference on Multimedia (ACM-MM'14)*, pages 1041–1044, Orlando, FL, USA, Nov. 2014.

[4] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8, 2015.

[5]