**CSCI 6908**-Deep Learning Summer 2018

**Project**: Audio Tagging

Submission date: 07/Ago/2018

Student ID: B00735030 - Student Name: Miria Rafante Bernardino

Contents

[Abstract 3](#_Toc519672490)

[Introduction 4](#_Toc519672491)

[Literature review 4](#_Toc519672492)

[Data Preparation 4](#_Toc519672493)

[Methods 4](#_Toc519672494)

[Results 5](#_Toc519672495)

[Conclusion 6](#_Toc519672496)

[References 7](#_Toc519672497)

## Abstract

## Introduction

The audio tagging problem can englobe different subproblems each with great variance of the complexity and difficulty each solution requires. Tagging environmental audios can be one of the most difficult once it must deal with noises and sounds that are not of interest and detect, above everything, when the sound of interest starts happening [5].

The problem focus of this work is a restricted and simplified problem. The sounds were recorded completely noiseless and were tagged into forty-one classes allowing the application of supervised machine learning techniques [1].

## Literature review

Automatic tagging has been a desired solution as showed by the DCASE - Detection and Classification of Acoustic Scenes and Events [1]’s challenges, that have worked this problem for a few years already, each year with a subtle different focus [5][6] (<http://www.cs.tut.fi/sgn/arg/dcase2017/>) which have contributed to motive several works on the area.

Hamel et al [9]’s work, for instance, compared mel-spectrum and MFCC, two forms to represent the sounds that will be explained in the Describing the data section, on building models for classification. They conclude that mel-spectrum performs better than MFCC and the use of PCA also improves the results. After stablishing this first steps, they compared pooling functions over time, variating the size of the windows and combining some of these functions. The pooling functions were mean, variance, maximum, minimum, etc., and they observed the best outcome came from the combination of the four mentioned pooling functions, besides the performance were greater with windows lengths around two to four seconds. Their final-best result was obtained by adding a hidden layer before the pooling step.

Lee et al [8], accordingly, compared STFT, MFCC and MFCC spectrum and observed the best approach was the one using MFCC spectrum. For their more outstanding approach, they used mel-spectrum to describe the data, with six convolutions by layer, each convolution followed by a max-pooling function. The mel-spectrum data were normalized by subtracting the average and dividing by its standard deviation. The filter length was fixed and had size of 243 values with stride of 81. They also used batch normalization and ReLU activation for the hidden layers and sigmoid for the output layer. The cost function used was cross entropy. In the last convolution layers’ output, they applied dropout of 0.5, stochastic gradient descent and 0.9 of momentum. The initial learning rate was 0.01 and decrease by 5 at each 3 epochs without improvement. The batch size was 23 for one database and 50 for the other. Their algorithm overcame prior state-of-the-arts methods and reach an AUC curve of 0.9059 for the same database used by Hamel et al [9], MagnaTagATune dataset, and Hamel performance was under 0.88.

Baseline structure as a previous literature

**System description**

**The baseline system implements a convolutional neural network (CNN) classifier similar to, but scaled down from, the deep CNN models that have been very successful in the vision domain. The model takes framed examples of log mel spectrogram as input and produces ranked predictions over the 41 classes in the dataset. The baseline system also allows training a simpler fully connected multi-layer perceptron (MLP) classifier. The baseline system is built on TensorFlow.**

**<strong>Input features (audio description was used)</strong>**

**We use frames of log mel spectrogram as input features:**

**\*computing spectrogram with a window size of 25ms and a hop size of 10ms**

**\*mapping the spectrogram to 64 mel bins covering the range 125-7500 Hz**

**\*log mel spectrogram is computed by applying log(mel spectrogram + 0.001)**

**\*log mel spectrogram is then framed into overlapping examples with a window size of 0.25s and a hop size of 0.125s**

**<strong>Architecture</strong>**

**The baseline CNN model consists of three 2-D convolutional layers (with ReLU activations) and alternating 2-D max-pool layers, followed by a final max-reduction (to produce a single value per feature map), and a softmax layer. The Adam optimizer is used to train the model, with a learning rate of 1e-4. A batch size of 64 is used.**

**The layers are listed in the table below using notation Conv2D(kernel size, stride, # feature maps) and MaxPool2D(kernel size, stride). Both Conv2D and MaxPool2D use the SAME padding scheme. ReduceMax applies a maximum-value reduction across the first two dimensions. Activation shapes do not include the batch dimension.**

**Layer Activation shape**

**Input (25, 64, 1)**

**Conv2D(7x7, 1, 100) (25, 64, 100)**

**MaxPool2D(3x3, 2x2) (13, 32, 100)**

**Conv2D(5x5, 1, 150) (13, 32, 150)**

**MaxPool2D(3x3, 2x2) (7, 16, 150)**

**Conv2D(3x3, 1, 200) (7, 16, 200)**

**ReduceMax (1, 1, 200)**

**Softmax (41,)**

**<strong>Clip prediction</strong>**

**The classifier predicts 41 scores for individual 0.25s-wide examples. In order to produce a ranked list of predicted classes for an entire clip, we average the predictions from all framed examples generated from the clip, and take the top 3 classes by score.**

**<strong>System performance</strong>**

**The baseline system trains to achieve an MAP@3 of ~0.7 on the public Kaggle leaderboard after ~5 epochs of the entire training set which are completed in ~12 hours on an n1-standard-8 Google Compute Engine machine with a quad-core Intel Xeon E5 v3 (Haswell) @ 2.3 GHz.**

**######**

## Data Preparation

### Dataset

The dataset is composed by 9473 samples unequally distributed among 41 categories. According to DCASE [1], “the minimum number of audio samples per category in the train set is 94, and the maximum 300. The duration of the audio samples ranges from 300ms to 30s ...”. Considering the rate is 44.1 kHz, 44100 values per second, and that the recording time is different for each sample, the number of points per sample variety drastically. The greater sample has 1323000 values.

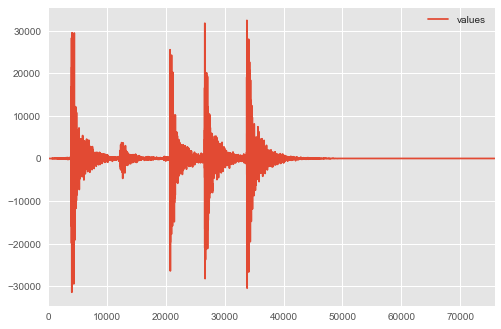


Figure 1 - Plotting a Knock sound sample. The x-axis is time in ms.

Figure 1 and Figure 2 show two samples’ values (knock and a oboe) over time (ms).

Some of the samples’ label were manually verified and some were not. For treat this problem we are using only the verified samples, in other words, we are using only the 3710 samples that were verified manually.

### Sounds properties

Sounds are waves propagating over a physical matter [3]. The height (or amplitude) of the waves and the number of waves flowing by second (or frequency) are examples of the properties of a sound [2]. The higher the amplitude, the more energy the wave has, and intensity is the unity to measure the amount of energy a wave has in a given area [2]. Besides, tones, overtones, harmonics, speed of sound, timbre, loudness, etc., are other properties of the sounds and they variety according to its source [2][3].

Therefore, to identify the source of a sound, we need to study how the waves are changing over time and we also need a minimum period of time to be able to observe such changes [3], and we did that by describing the samples in function of some of its properties.

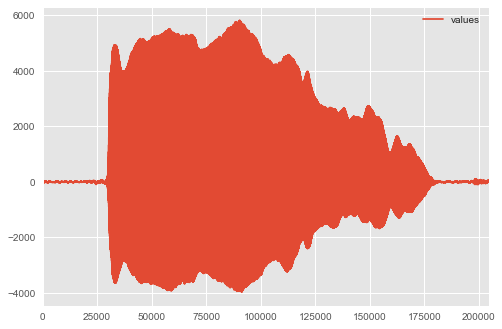


Figure 2 - Plotting a Oboe sound sample. The x-axis is time in ms.

### Describing the data

We used the 4 functions shown in Figure 3 and Figure 4 from librosa [4] package for feature extraction to describe the original data: tonnetz (computes the tonal centroid features (tonnetz)), spectral\_centroid (computes a 6D description of chords), spectral\_bandwidth and mfcc (computes the Mel-frequency cepstral coefficients (MFCCs)).

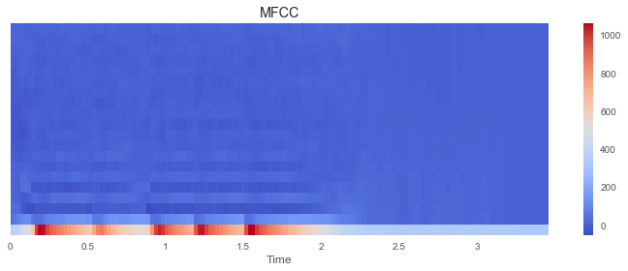
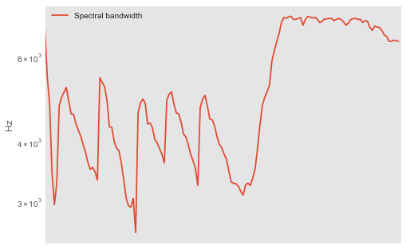
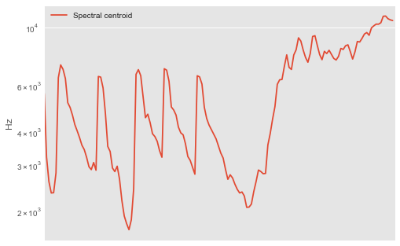
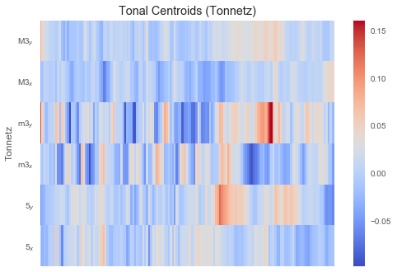


Figure 3 - Vizualization of the knock sample trough librosa's functions for feature extraction: (top-left) Tonnetz, (top-right) Spectral centroid, (bottom-left) MFCC and (bottom right) Spectral bandwidth. Source: https://librosa.github.io/librosa/index.html

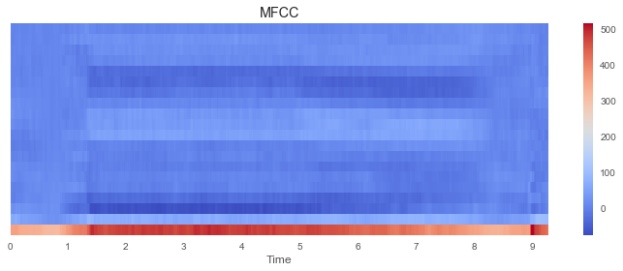
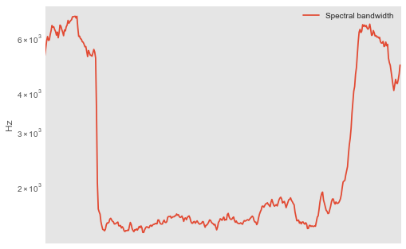
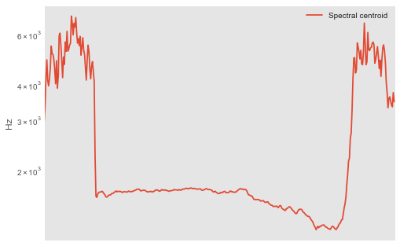
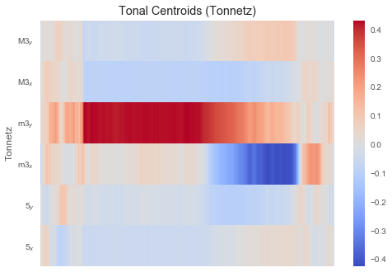


Figure 4 - Visualization of the oboe sample trough some librosa's functions for feature extraction: (top-left) Tonnetz, (top-right) Spectral centroid, (bottom-left) MFCC and (bottom right) Spectral bandwidth. Source: https://librosa.github.io/librosa/index.html

####Explain a little bit a about each function and mention the format of the output and what do they mean###

Besides, we tried a fifth description trough spectrograms. The function is the stft function fro the librosa [4] package and creates a 3-dimensional space combining the time series, all the frequencies and the intensity at each point of time at each frequency.

Padding – paste the graphics here

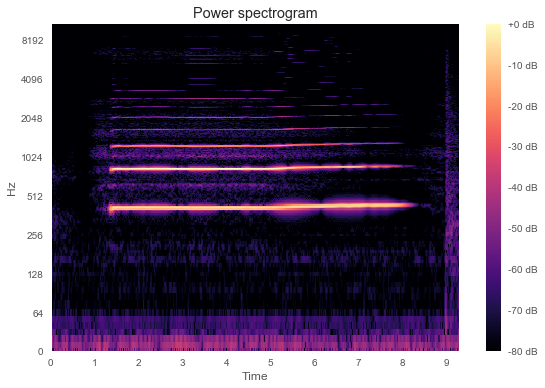
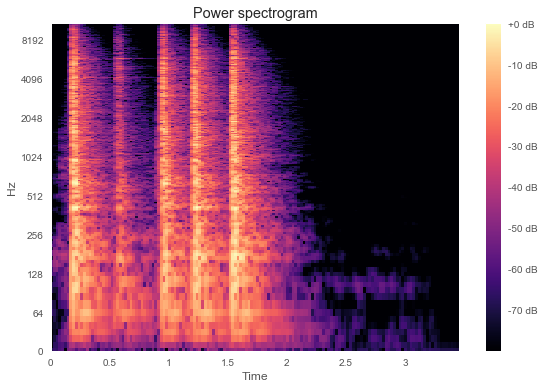


Figure 5 - Power spectrograms of a Knock and an Oboe sound sample

## Methods

## Results

|  |  |  |  |
| --- | --- | --- | --- |
| Data description | MaxSize | RF – 3-fold | MLP- 3-fold |
| Tone | 1000 | Scores: [0.2971246, 0.31012146, 0.29681112]  Mean: 0.30135239277503162  Std: 0.0062019856754040002 | Scores: [0.0686901, 0.06720648, 0.06950123]  Mean: 0.068465933357223716  Std: 0.00095014194851230651 |
| Spectral Centroid | 1000 | Scores: [0.43210863, 0.43724696, 0.4480785]  Mean: 0.4391446950878993  Std: 0.006656336335106039 | Scores: [ 0.0686901, 0.06882591, 0.0678659]  Mean: 0.06846063676458795  Std: 0.00042417933477714254 |
| Spectral Bandwidth | 1000 | Scores: [ 0.44888179, 0.43076923, 0.43254293]  Mean: 0.4373979823782462  Std: 0.008152499127789991 | Scores: [ 0.0686901, 0.06720648, 0.06950123]  Mean: 0.06846593335722372  Std: 0.0009501419485123065 |
| MFCC | 1000 | Scores: [0.64616613, 0.65101215, 0.65494685]  Mean: 0.65070837731252074  Std: 0.0035911426034004267 | - |
|  |  |  |  |

|  |  |  |
| --- | --- | --- |
| Data description | MaxSize | RF – 10-fold |
| MFCC | 1000 | Scores: [0.67792208, 0.66318538, 0.67810026, 0.63852243, 0.7037037, 0.67204301, 0.67847411, 0.66111111, 0.64145658, 0.69714286]  Mean: 0. 67116615275898628  Std: 0. 019986117622205764 |

## Conclusion

## References

1. Dcase. "General-purpose Audio Tagging of Freesound Content with AudioSet Labels." General-purpose Audio Tagging of Freesound Content with AudioSet Labels - DCASE. Accessed July 19, 2018. http://dcase.community/challenge2018/task-general-purpose-audio-tagging.
2. "The Components of Sound." Conductors and Insulators. Accessed July 19, 2018. <https://www.nde-ed.org/EducationResources/HighSchool/Sound/components.htm>.
3. "Sound." Wikipedia. July 15, 2018. Accessed July 19, 2018. <https://en.wikipedia.org/wiki/Sound>.
4. "LibROSA." LibROSA - Librosa 0.6.0 Documentation. Accessed July 19, 2018. <https://librosa.github.io/librosa/index.html>.
5. S. Yun, S. Kim, S. Moon, J. Cho, and T. Kim, “Discriminative training of gmm parameters for audio scene classification and audio tagging,” IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE 2016).
6. H. Eghbal-Zadeh, B. Lehner, M. Dorfer, and G. Widmer, “CP-JKU submissions for dcase-2016: A hybrid approach using binaural i-vectors and deep convolutional neural networks,” Tech. Rep., DCASE2016 Challenge, Tech. Rep., 2016.
7. Choi, K., Fazekas, G., & Sandler, M. (n.d.). AUTOMATIC TAGGING USING DEEP CONVOLUTIONAL NEURAL NETWORKS. Retrieved from <https://arxiv.org/pdf/1606.00298.pdf>
8. Lee Jiyoung Park Keunhyoung Luke Kim Juhan Nam, J. (n.d.). SAMPLE-LEVEL DEEP CONVOLUTIONAL NEURAL NETWORKS FOR MUSIC AUTO-TAGGING USING RAW WAVEFORMS. Retrieved from <https://github.com/keunwoochoi/MSD_split_for_>
9. Hamel, P., Lemieux, S., Bengio, Y., & Eck, D. (n.d.). TEMPORAL POOLING AND MULTISCALE LEARNING FOR AUTOMATIC ANNOTATION AND RANKING OF MUSIC AUDIO. Retrieved from http://www.iro.umontreal.ca/~lisa/bib/pub\_subject/finance/pointeurs/music\_pooling.pdf
10. Xu, Y., Kong, Q., Huang, Q., Wang, W., & Plumbley, M. D. (2017). Convolutional gated recurrent neural network incorporating spatial features for audio tagging. In 2017 International Joint Conference on Neural Networks (IJCNN) (pp. 3461–3466). IEEE. <https://doi.org/10.1109/IJCNN.2017.7966291>