Urban Sound Classification with NN

Martina Casati 942413

Gennaio 2021

Contents

Abstract

The aim of this project is to train neural network for the classification of audio files by using Tensorflow. This work wants to explore different neural network architectures, looking for the best one that solves the problem of audio recognition. In particular, for achiving this purpose, Neural Network will be implemented using two different feautures extracted from the audio files.

1 Dataset

The UrbanSound8K dataset is a collection of 8732 short (up to 4 seconds) audio clips of urban sound areas. And the audio clips are prearranged into 10 folds. The dataset is divided into 10 classes:

- air conditioner = 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

On top of data, the included metadata file provides all the required information about each audio file:

- slice file name: The name of the audio file.
- fsID: The Freesound ID of the recording from which this excerpt (slice) is taken
- start: The start time of the slice in the original Freesound recording
- end: The end time of slice in the original Freesound recording
- salience: A (subjective) salience rating of the sound. 1 = foreground, 2 = background.
- fold: The fold number (1-10) to which this file has been allocated.
- classID: A numeric identifier of the sound class.

- class: The class label name.

	slice_file_name	fsID	start	end	salience	fold	classID	class
0	100032-3-0-0.wav	100032	0.000000	0.317551	1	5	3	dog_bark
1	100263-2-0-117.wav	100263	58.500000	62.500000	1	5	2	children_playing
2	100263-2-0-121.wav	100263	60.500000	64.500000	1	5	2	children_playing
3	100263-2-0-126.wav	100263	63.000000	67.000000	1	5	2	children_playing
4	100263-2-0-137.wav	100263	68.500000	72.500000	1	5	2	children_playing
5	100263-2-0-143.wav	100263	71.500000	75.500000	1	5	2	children_playing
6	100263-2-0-161.wav	100263	80.500000	84.500000	1	5	2	children_playing
7	100263-2-0-3.wav	100263	1.500000	5.500000	1	5	2	children_playing
8	100263-2-0-36.wav	100263	18.000000	22.000000	1	5	2	children_playing
9	100648-1-0-0.wav	100648	4.823402	5.471927	2	10	1	car_horn
10	100648-1-1-0.wav	100648	8.998279	10.052132	2	10	1	car_horn

2 Exploring the data

An audio file is presented as a numpy array, where each entry is a discretized sample having a sampling rate of 44100 Hertz. Figure 1 is an example of sound wave from raw data.

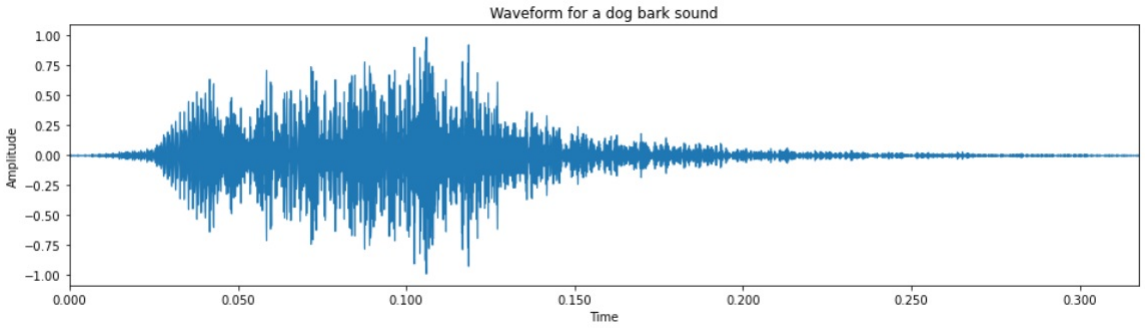
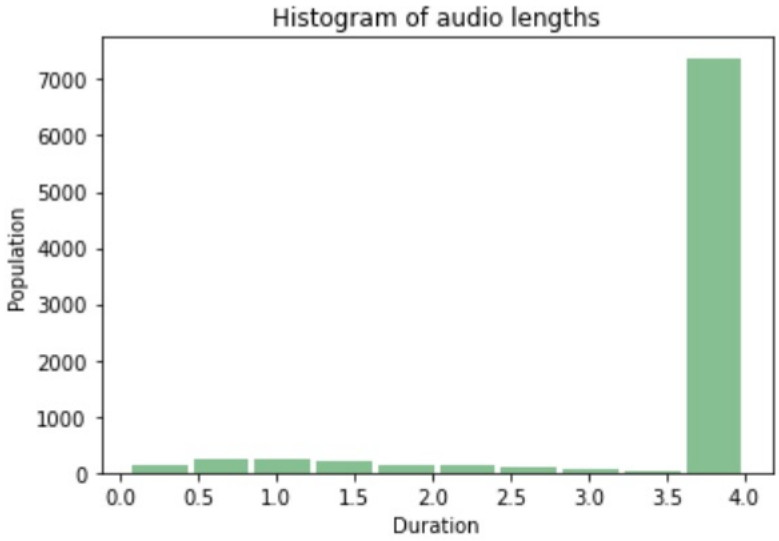


Figure 1: Sample of sound wave of dog bark

Each audio file approximately has a duration of four seconds, as shown in the figure 2. Most samples have a duration greater than 3 seconds. Only the 14.4% of the dataset has less than 3 seconds and 8.5% is lower than 1.5 seconds.




```
Greater than 3 seconds: 7468
Lower than 3 seconds: 1264
Lower than 1.5 seconds: 785
```

Figure 2: Histogram of audio lenghts

The dataset is given with ten folds, and the models will be trained on folds 1, 2, 3, 4, 6 and will be tested on folds 5, 7, 8, 9, 10.
Each fold has, on average, an equally number of sound files, as shown in the figure below.

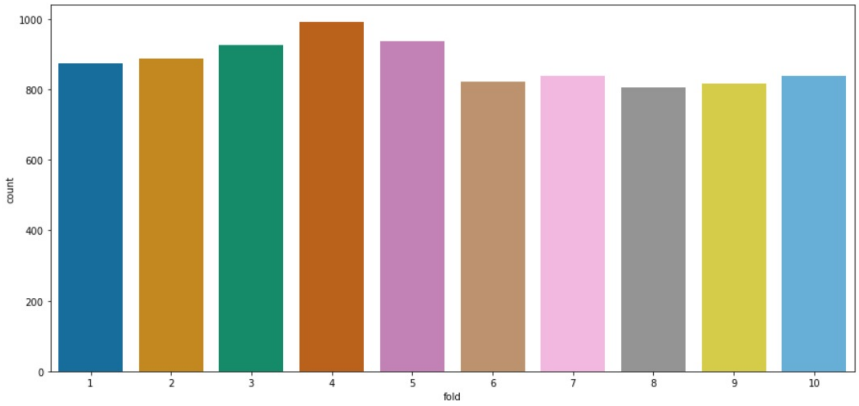


Figure 3: fold distribution

As shown in the graph below, the population is well distributed among 8 classes but we observe a significant difference between those and the two last classes: "car horn" and "gun shot".

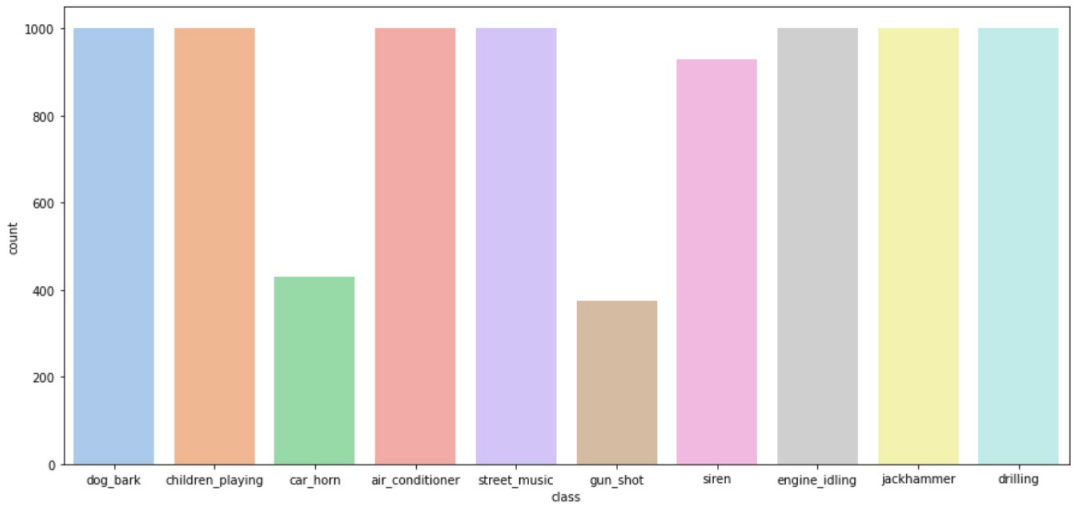


Figure 4: class distribution

In conlusion, the dataset is not fully balanced in means of classes, with two classes containing less than then 50% of the mean per-class count of the dataset. This may or may not be a problem depending on the complexity of the patterns that describe the occurrences belonging to those classes.

3 Features extraction

In this work, two different type of features have been extracted:

- Mel-frequency cepstral coefficients(MFCCs),which has been the most common approach for the last years as it has proved to perform much better than normal spectrograms. MFCC takes into account human perception for sensitivity at appropriate frequencies by converting the conventional frequency to Mel Scale. 40 coefficients (also referred as bands) per frame have been extracting with the librosa.feature.mfcc() function.

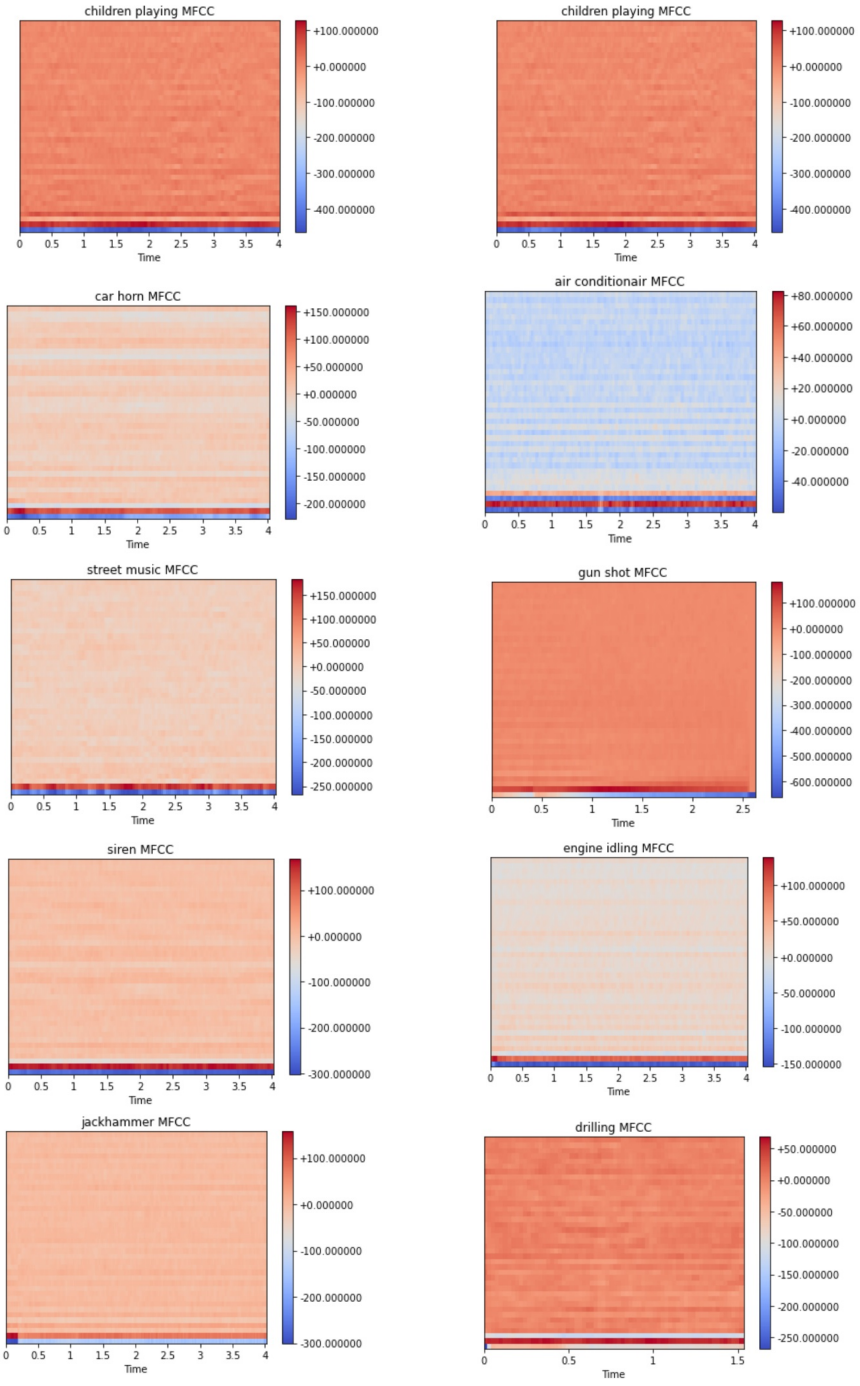


Figure 5: MFCCs of the different classes of sound

By looking at the figure 5 it is possibile to observe the Mel-frequency cepstral coefficients for each category of the urban sound dataset, showing the different characteristics that each class present.

- Chromagram, closely relates to the twelve different pitch classes. Chroma-based features, which are also referred to as "pitch class profiles", are a powerful tool for analyzing music whose pitches can be meaningfully categorized (often into twelve categories) and whose tuning approximates to the equal-tempered scale. One main property of chroma features is that they capture harmonic and melodic characteristics of music, while being robust to changes in timbre and instrumentation. 40 coefficients (also referred as bands) per frame have been extracting with the librosa.feature.chroma_stft() function.

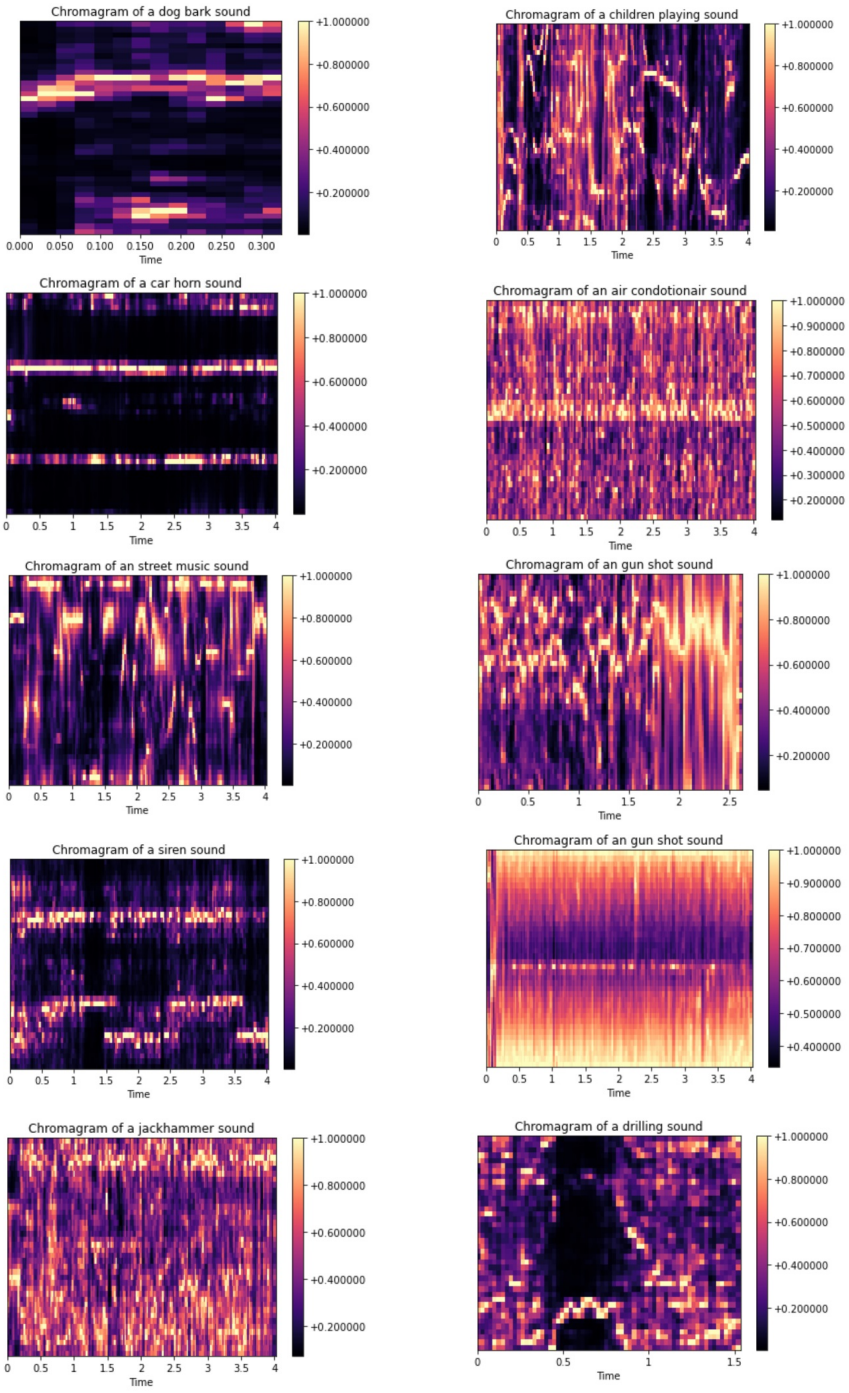


Figure 6: Chromagram of the different classes

By looking at the figure 6 it is possibile to observe the Chromagram for each category of the urban sound dataset, showing the different characteristics that each class present.

4 Model implemented

In this project three different network architectures have been implemented:

- MODEL1 as the simplest neural network, composed by only two hidden layer, respectively with 512 and 128 neurons.
- MODEL2 as intermediate model, having two hidden layer with 512 and 128 neurons each, as before, and dropout layer; The archiecture is structured as the previuos one but enriched with a regulation technique, in order to try to keep overfitting issue under control. The "Dropout rate" is the fraction of features that are being zeroed-out , which is set in my work to 0.3 (usually fix between 0.2 and 0.5). Adding dropout layer should improve the performance of the model. Model 2 is structured as follow:

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_8 (Dense)             (None, 128)               5248

 dropout_6 (Dropout)         (None, 128)               0

 dense_9 (Dense)             (None, 512)               66048

 dropout_7 (Dropout)         (None, 512)               0

 dense_10 (Dense)            (None, 128)               65664

 dropout_8 (Dropout)         (None, 128)               0

 dense_11 (Dense)            (None, 10)                1290

=================================================================
Total params: 138,250
Trainable params: 138,250
Non-trainable params: 0
```

Figure 7: Model 2 architecture

- MODEL3, where dropout layers are combined with batch normalization, in order to increase more performace and stability of the neural network. In short, it normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation. This process optimizes training time. Model 3 is structured as follow:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 512)               20992

 dropout (Dropout)           (None, 512)               0

 batch_normalization (BatchN  (None, 512)              2048
 ormalization)

 dense_1 (Dense)             (None, 128)               65664

 dropout_1 (Dropout)         (None, 128)               0

 batch_normalization_1 (Batc  (None, 128)              512
 hNormalization)

 dense_2 (Dense)             (None, 128)               16512

 dropout_2 (Dropout)         (None, 128)               0

 batch_normalization_2 (Batc  (None, 128)              512
 hNormalization)

 dense_3 (Dense)             (None, 10)                1290

=================================================================
Total params: 107,530
Trainable params: 105,994
Non-trainable params: 1,536
```

Figure 8: Model 3 architecture

4.1 Activation and loss function

In all layers exept for the last one the Relu (rectified linear activation function) activation function has been used, which is the most commonly used activation function in deep learning models. The function returns 0 if it receives any negative input, but for any positive value x it returns that value back. So it can be written as f(x)=max(0,x).
Graphically it looks like this

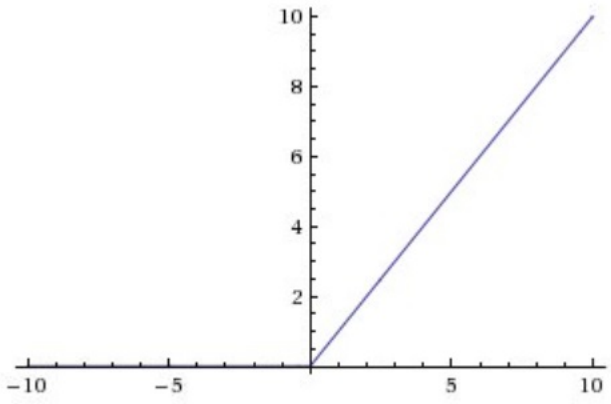


Figure 9: Relu activation function

The Softmax activation function has been adopted in the last layer. The softmax function is used to normalize the outputs, converting them from weighted sum values into probabilities that sum to one. Each value in the output of the softmax function is interpreted as the probability of membership for each class. In all the architectures implemented, it has been decided to use the *Categorical Crossentropy* as a loss function for training the algorithm. The optimazer used to train the network *Adam*.

5 Analysis

First of all, we perform a 10-fold cross-validation for all the models, for both mfcc and chromagram features. The results in terms of accuracy and standard deviation are shown in figures 10 and 11 in a way which allow us to interpret easily the meaning.

	Accuracy	St.Err.
Model1	91.79%	1.62%
Model2	90.66%	1.02%
Model3	90.86%	1.03%

Figure 10: results 10-cv for mfcc

	Accuracy	St.Err.
Model1	70.56%	1.5%
Model2	70.28%	1.47%
Model3	66.88%	1.77%

Figure 11: results 10-cv for chromagram

As we can see, for the MFCCs features, there is a good balance between the model2, with dropout layer, and the model3 ,composed by the combination of the dropout layer with batch normalization. Contrary to expectation, for Chromagram features, we can see that there is a good balance between the simplest model1 and the intermediate model 2.

6 Results

After having performed a 10-fold cross-validation,we train our model2 and model3 over the folds number 1,2,3,4,6 for the extracted features MFCCs and we also test them over the folds 5,7,8,9 and 10. Similarly, we train models 1 and 2 over the folds 1,2,3,4 and 6 for the Chromagram and test them over the remainimg folds. Then, we calculate the average accuracy of each model tested on each fold for each of the two features, having the following resuts:

	MFCC Accuracy	Chroma Accuracy
Model1	37.5%	38.4%
Model2	41.2%	38.7%
Model3	40.2%	0%

Figure 12: Average accuracy

As we can see, the model2 has the highest value for average accuracy for both the extracted features. Nevertheless, the accuracy's values are respectively 41.2% and 38.7% for MFCCs and Chromagram, which are relatively low values:it means that when we extract MFCCs and Chromagram from audio files and we use model 2 to predict and classify data, it correctly outputs on average less than the half of the results.
In addiction, it could be interesting to analyse the behaviour of our models with respect to the training and validation accuracy, focusing on overfitting and underfitting issue.

- MFCC:
  - The results in the figure 13 are obtained by using model2 architecture on the Mfcc's features.
    As we can see, train and validation are closer to each other compared to the Model3 model in the next point. Furthermore, the evolution of the test accuracy as function of number of epochs is more stable suggesting higher robustness. This behaviour means that the model is capturing the data more effctively, reducing overfitting.

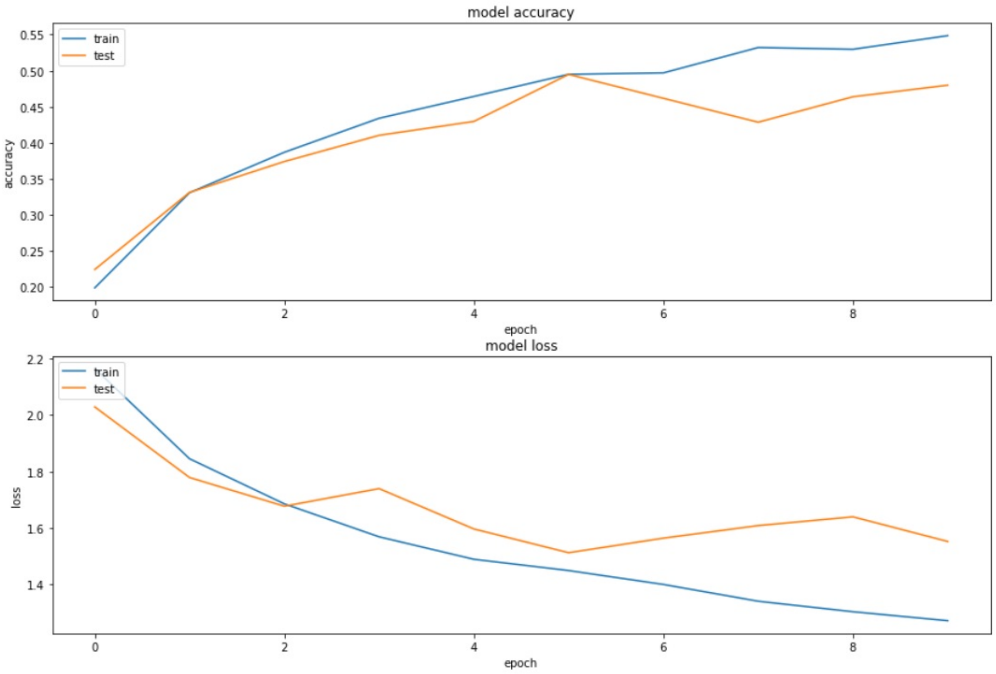


Figure 13: Accuracy and loss for training and validation of model2

– The result of figure 14 are obtained by using Model 3 architecture, with dropout layer and batch normalization.
Contrary to expectations, adding batch normalization ,results in a worse performance that the previous model on the test set and loss and accuracy appear less stable as the number of epochs increse.

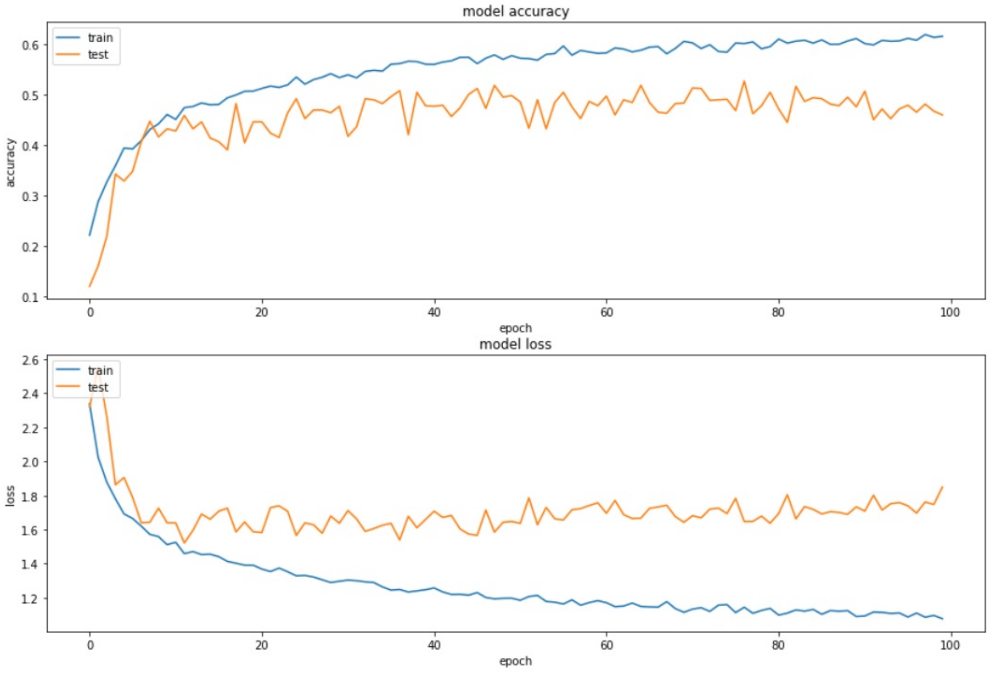


Figure 14: Accuracy and loss for training and validation of model3

.

- CHROMAGRAM:
  - The result of figure 15 are obtained by using model2 over Chromagram features. As we can see, the modelis overfitting because the accuracy measured against the training set is very good, while the accuracy measured against the validation set assumes low value.

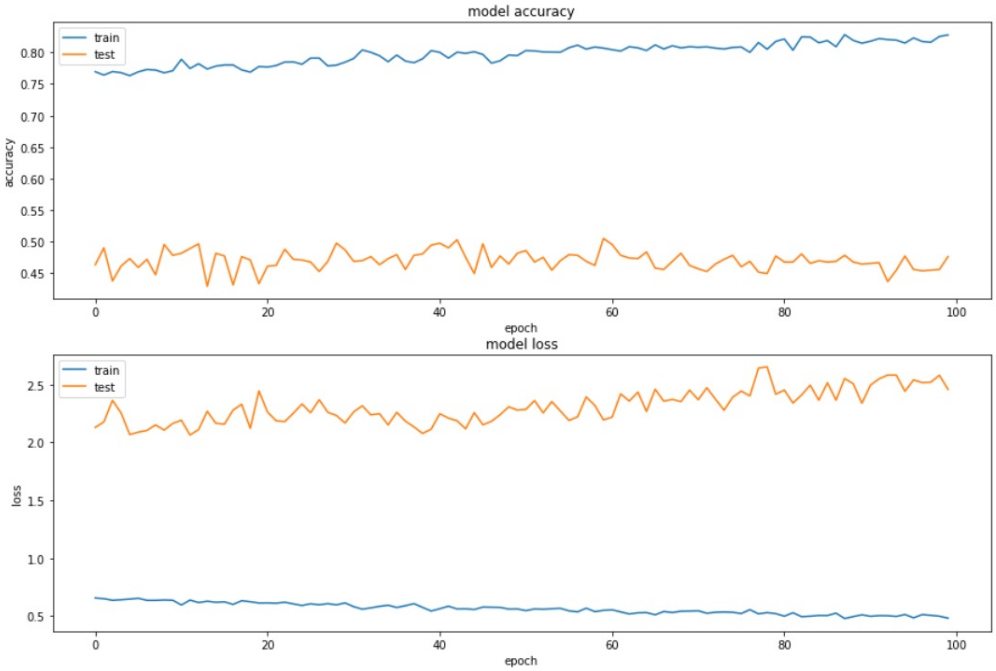


Figure 15: Accuracy and loss for training and validation of model2

7 Conlcusion

After our analysis we can conlcude that, regardless the partition of the dataset, MFCC's feature always outclasses the Chromagram. This results could mean that Mfcc features better explain the characteristics of Urband Sound from a classification point of view; Indeed, one main property of chroma features is that they capture harmonic and melodic characteristics of music, and probably in our case we don't have categories closely related to "music".
That could be the reason why Chromagram performed worse that MFCC.

8 Declaration

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/our 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.