Urban Sound Classification with Neural Networks

Statistical Methods for Machine Learning project report

Priscilla Raucci*
Università degli Studi di Milano, Via Celoria 18, Milan, Italy
Email: *priscilla.raucci@studenti.unimi.it

***Abstract*—In this study, two models of neural networks are developed for the classification of sound events based on audio files from the UrbanSound8K dataset. The first model takes advantage of the robust machine learning techniques developed for image classification. It is a CNN that taking in input the Mel spectrogram image tries to label audio excerpt. The second model that was developed and analysed is a classical Feedforward neural network using as input MSCCs statistics extracted from the previously cited audio dataset.**

Using these self-developed architectures was achieved an average f1-score of 74% (78% for the CNN, 70% for the Feedforward) in classifying the top 5 classes. The predictions of the two models were applied to 10 classes.

***Index Terms*—Urban Sound Sound Classification, Neural Networks, CNN, Deep learning**

I. Introduction

Automatic sound recognition is a well-known research area that has been widely explored in the last years. There is a large body of research related to different applications of this problem, such as speech [1], [2], music [3], [4], [5] and bioacoustics, while less is available on the analysis of urban acoustics environments is relatively few. The automatic environmental sound classification is the task of identifying automatically audio excerpts recorded from the environment and labelling them with pre-defined classes.

Recognizing sounds automatically is crucial for several potential applications such as context-aware computing, surveillance and information retrieval techniques.

The aim of this work is to develop Neural Networks capable of the classification of audio files from the UrbanSound8K dataset [6].

In this work two Neural networks are proposed, both based on the use of features extracted from the Mel spectrogram. The Mel spectrogram is a representation of audio records that depicts frequency composition of the audio signal over time. The use of Mel spectrograms in audio classification is widely recognised and its use is preferred to other signal representations such as the chromatogram for its reduced size. [5], [7].

The first Neural network proposed is a Convolution neural network (CNN) that takes as input the Mel spectrogram image.

CNNs are a special kind of neural networks primarily used for image classification tasks [8], [9]; over time they have seen a lot of successful applications in many different domains. CNNs are commonly and successfully used in Time Series Classification (TSC)[10], a problem which sounds classification in a part of. In recent years this technique was applied to several audio recognition fields such as speech [2], music [**?**] and environment sound analysis [11], [12], [13], achieving promising results.

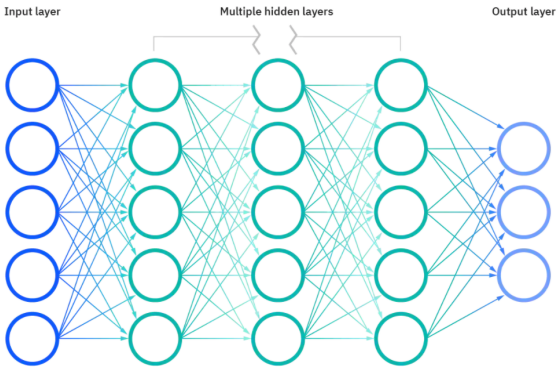


Figure 1. Deep Neural Network scheme.

This Model was then compared to a second Neural Network; the second model is a standard Neural network that takes as input a series of statistics extracted from the Mel-Frequency Cepstral Coefficients (MFCC) of each audio excerpt. MFCCs are commonly used in environmental sound analysis and are often used as a benchmark to test model performances[6].

This second method is more common and standardised than the previous CNN description. Its aim was to compare CNN performances with a simpler method to be used as a baseline.

In this study, the audio files were labelled using the 10 classes predefined in the given dataset: air conditioner, car horn, children playing, dog bark, drilling, engine idling, gunshot, jackhammer, siren, and street music.

The rest of the paper is organized as follows. In Section II an introduction is given to the main techniques and models used. In Section III the dataset and preprocessing methods are explained in details and in Section IV a detailed description of developed models is given. Section V contains the results for both approaches and finally Section VI concludes the paper.

II. Proposed Methods

In the problem of automatic environmental sound classification a variety of different learning techniques were used to perform the task, Deep neural networks are widely and successfully used techniques.

In this work were developed and compared two main Neural structures. First were a Convolutional Neural Network was developed to solve the urban sound classification problem,

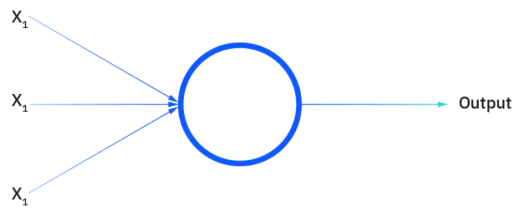


Figure 2. Representation of an artificial Neuron.

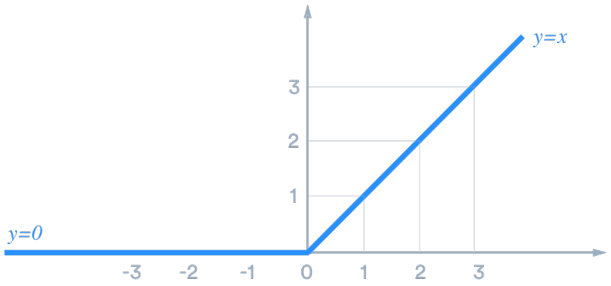


Figure 3. ReLu activation function .

in the second instance, was a Multi-Layer Perceptron neural network (or Feedforward neural network) was developed to classify the same dataset in order to compare performances.

A. Neural Networks

Neural networks are a subset of machine learning and are the main component of deep learning algorithms. These networks are inspired by the human brain and they simulate the information flow across neurons.

A neural network is a composition of parametric functions referred to as layers. Each layer contains neurons, which are small units that compute one element of the layer's output. In Neural networks, layers can be addressed as the input layer, hidden layer, or output layer, according to their position in the network, as depicted in Figure1. A Deep Neural network is a Neural Network characterized by multiple hidden layers.

Each layer takes as input the output of its previous layer and computes its own output, according to the parameters of weight and threshold associated with its neurons.

B. Neurons' sctivaction function

Each individual neurons uses an activaction function to determinate the output value. The neuron computes a linear combination of its inputs and gives it to an activation function.

The activation function is used to introduce non-linearity in the neural network. Traditionally, logistic sigmoid and hyperbolic tangent have been used as typical non-linear activation functions in a multilayer perceptron setup. More recently, these functions have been replaced by alternatives solutions. One of the most common is the Rectified Linear Units (ReLUs), which follows the following activation function

$$ReLu(x) = \max(0, x).$$

. The main advantages of ReLu over traditional units is the faster computation and more efficient gradient propagation while it remains a quite simple function.

An other common activaction funcion is the so-called Softmax activation function. It is used mainly in output layers' neurons in Neural networks that solve classification problems. The Softmax function is very useful because it converts scores to a normalized probability distribution. The Softmax formula is the following

$$Softmax(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

where all the z_i values are the elements of the input vector, they can take any real value. e^{z_i} is the standard exponential function, it is applied to each element of the input vector. This gives a positive value above 0. The term on the bottom of the formula is the normalization term. It ensures that all the output values of the function will sum to 1 and each be in the range (0, 1), thus constituting a valid probability distribution. K is the number of classes in the multi-class classifier.

C. Neural networks' training

A crucial phase of the Neural Network developing process is the training phase. Neural Networks effectiveness relies on the training data to learn and improve their performances. During the training the network tune neuron parameters to optimize their ability to solve the given task.

The ability of the Neural Network to perform satisfying its task is commonly measured using the accuracy metrics. The accuracy is the proportion of the total number of correct predictions. Some other commonly used metrics are precision, recall and f1-score. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations, recall is the ratio of correctly predicted positive observations to all observations in actual class, f1-score is the weighted sum of precision and recall.

During the training phase, the weights are initialised randomly, an input x is computed layer by layer until the output layer. The output is a vector whose components are the estimated probabilities of x belonging to each class.

As the model is trained a cost (or loss) function is used to evaluate the model's prediction accuracy. Ultimately, the goal is to minimise the cost function to ensure the correctness of the model.

The weights are adjusted and updated using the so-called gradient descent function. This function permits the model to determine the direction to take to reduce errors. This process is repeated iteratively over all the training data until the model's parameters are updated in a way that minimises the loss on the training data.

This kind of training produces a supervised learning model, as the model is trained with data that are labelled. The training datasets are designed to train or "supervise" algorithms.

Supervised learning is in contrast with unsupervised learning, whose aim is to produce algorithms capable of discovering "hidden patterns" into data without human intervention.

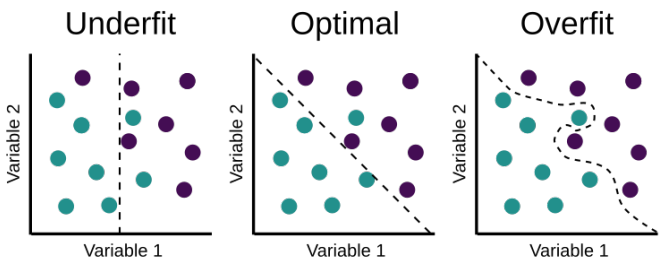


Figure 4. Example of overfit and underfit models for classification task with two classes.

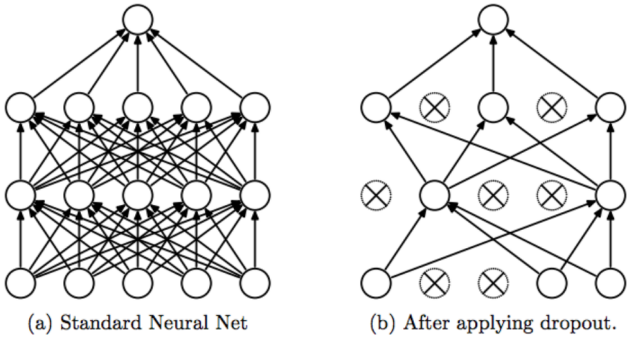


Figure 5. Sample of a Dropout Neural network model. On the left, there is a standard NN with 2 hidden layers, on the right the same network using the Dropout method. Crossed units have been dropped.

The two models proposed in this work are both supervised learning models.

D. Evaluation of a NN

The ultimate performance of a Neural Network model is evaluated over the testing data. Testing data are new data not used in the training phase. During the training phase the model should learn from examples of the training set and generalise from these examples to new data. This process is hard to perform: the two main ways of failing it are underfitting and overfitting.

An underfit model is a network that performs poorly on the training set and does not perform well on the testing set. An Overfit model is a model that performs very well on the training dataset but its error on the testing set is very high.

In Figure 4 is depicted the behaviour of overfit and underfit models for a two classes classification task.

One way to tackle the overfitting problem is to introduce a Dropout perturbation. In each training iteration, every hidden unit is randomly removed with a predefined probability (originally 50%), and the learning procedure continues normally. In Figure 5 is represented an example of this procedure. These random perturbations prevent the model from learning false dependencies and reduce dependency between neurons on the same layer. The Dropout technique tries to ensure each hidden unit learn data features that are generally favourable in producing the correct classification answer.

Figure 6. Diagrma of k-fold validation for k=10.

E. Hyperparameters and K fold cross-validation

A central point of the performance of a Neural Network is the fine-tuning of its hyperparameters, special parameters whose value must be determined before the training phase(such as batch size, learning rate, dropout percentage). The setting of hyperparameters is crucial as the wrong decision can lead to overfitting.

K fold cross-validation is a statistical method used to estimate the skill of machine learning models and adjust hyperparameters. It is commonly used in machine learning to evaluate models and adjust properly hyperparameters.

The Cross-validation procedure has a single parameter called k that refers to the number of groups that a given data sample is going to be split into. Data are shuffled and divided into k folders, for each folder the following procedure is performed:

- the data in the folder are held out as the test data set
- the rest of the data are used as the training data set
- the model is fitted on the training set and evaluate it on the test set
- the evaluation score is memorised, the model is discarded

This process is repeated iteratively, each sample is used only 1 time as validation data and used k-1 times for the model training. At the end of the procedure, evaluations are summarised. In Figure 6 is depicted the main schema.

Although cross-validation is not often used in Neural Networks development because of the long time required in training the large number of parameters, it was used in this case because of the small number of samples in the training set.

F. Types of neural networks: Multi-Layer Perceptron

An Multi-Layer Perceptron(MLP) or Feedforward neural network constitutes the simplest and most traditional architecture for deep learning models. This form of architecture is composed of an input layer, one or more hidden layers, and an output layer.

In the MLP layer, neurons can be organised in several different ways. One kind of proposed layer in literature is the Fully Connected Layer which is, each neuron in one layer

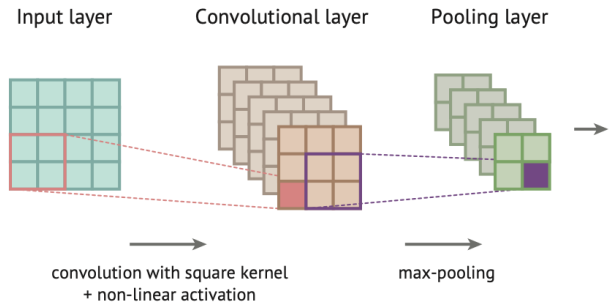


Figure 7. A schematic visualization of a typical convolution- pooling operation performed on the input data.

is connected to all neurons in the following layer. The full connectivity of these schemas makes them very rapid to adapt to training data but also prone to overfitting.

G. Types of neural networks: Convolutional Neural Network

Convolutional Neural Networks(CNNs) are essentially an extension of feedforward neural networks. These networks are usually used for image recognition and pattern recognition. They use basic principles from linear algebra, particularly matrix multiplication, to identify patterns within an image.

This architecture is composed of different layers stacked together: an input layer, a group of convolutional and pooling layers, some fully connected hidden layers, and an output (loss) layer.

The main difference from multilayer perceptrons is the introduction of convolution and pooling operations.

The convolutional layer introduces a new way of organising hidden neurons. Each Convolutional layer takes into account only part of the input, it focuses only on a small part of the whole input space, these tiny areas are called receptive fields. The weights of such a hidden unit create a convolutional kernel (filter) which is applied to (tiled over) the whole input space, resulting in a feature map.

The pooling layer is used to reduce the dimensionality of the following layers. The most commonly used pooling techniques are maxpooling and average pooling: maxpooling takes the maximum value of the pooling window and average pooling takes the average value.

III. Dataset

In order to train and test models was used UrbanSound8K [6], a publicly available dataset that collects audio file records of urban sounds. Each audio file is labelled with one of the 10 possible classes: air conditioner, car horn, children playing, dog bark, drilling, engine idling, gunshot, jackhammer, siren and street music.

The UrbanSound8K dataset provides a metadata CSV file that contains information of each audio sample, as shown in Figure 8.

Each audio file is provided with the name and both numerical and textual class labels. Every excerpt was drawn from a larger dataset and the metadata records the start and stop

	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

Figure 8. First 10 rows of the metadata file.

	index	jackhammer	children_playing	street_music	dog_bark	drilling	air_conditioner	engine_idling	siren	car_horn	gun_shot
0	fold1	120	100	100	100	100	100	96	86	36	35
1	fold2	120	100	100	100	100	100	100	91	42	35
2	fold3	120	100	100	100	100	100	107	119	43	36
3	fold4	120	100	100	100	100	100	107	166	59	38
4	fold5	120	100	100	100	100	100	107	71	98	40
5	fold6	68	100	100	100	100	100	107	74	28	46
6	fold7	76	100	100	100	100	100	106	77	28	51
7	fold8	78	100	100	100	100	100	88	80	30	30
8	fold9	82	100	100	100	100	100	89	82	32	31
9	fold10	96	100	100	100	100	100	93	83	33	32

Figure 9. Distribution of labels in predefined folders.

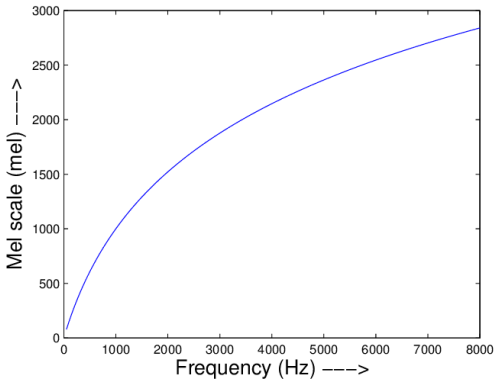


Figure 10. Plots of pitch mel scale versus Hertz scale.

time of the extracted piece of the original field recording as well as the original file ID. The salience feature refers to the importance of the sound in the recording, it is 1 when the sound is in the foreground, and 2 when the sound is in the background.

The dataset collects 8732 audio files of 4 seconds length, which are already arranged in 10 predefined folds. In this work folders 1, 2, 3, 4 and 6, were used as training set and folders 5, 7, 8, 9 and 10 were used as testing set. The class distribution in each fold is balanced as shown in Figure 9, as well it is not perfectly balanced, it is also not critically compromised.

A. Data Extraction

Audio files were read and elaborated using librosa[14] python library. Each audio file was first processed by trimming or zero-padding to set the length equal to 3.6 seconds. This is necessary because models require a static input dimension and 3.6 seconds is the average length of the audio data.

Following, the signal extracted was normalised and transformed into its Mel spectrogram representation.

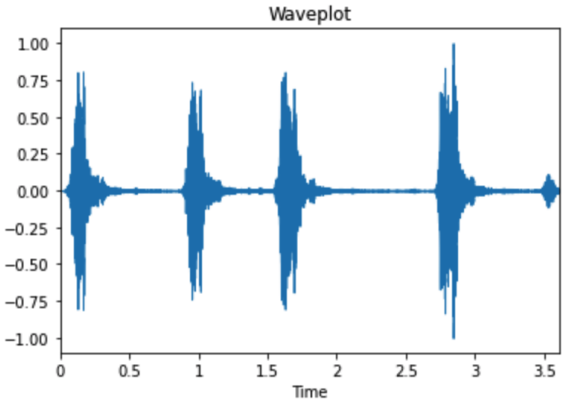


Figure 11. Waveplot representation.

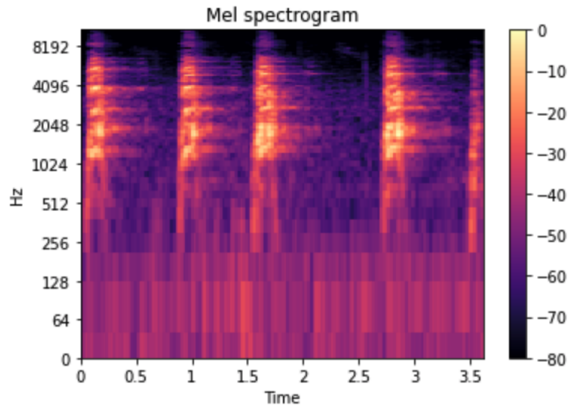


Figure 12. Melspectrogram representation.

A spectrogram is a visual depiction of a signal's frequency composition over time. The Mel spectrogram is a spectrogram that uses the so-called Mel-Scale on its y-axis. The Mel scale is the result of a non-linear transformation of the frequency scale, it is constructed to represent human auditory system in a linear way.

Human hearing does not perceive frequencies linearly: lower frequencies are easier to discern than higher ones. For instance, we can easily tell the difference between 500 and 1000 Hz, but we will hardly be able to tell a difference between 10000 and 10500 Hz, even though the distance between the two pairs are the same.

Briefly, the Mel spectrogram is able to depict the sound information as similar as possible to what a human would perceive.

The raw audio data was passed to the librosa Mel spectrogram extraction function, resampled to 22050, with window size of 1024, hop length of 512 and 60 mel-bands. After this process, each sample has the shape 128x156. In Figures 11 and 12 are represented a raw audio clip and the corresponding Mel frequency representation, respectively.

After this feature extraction phase, each mel-spectrogram data was stored, with its associated class label, in a csv file.

There are totally 10 csv files, in which data are organised according to the predefined folds of the original dataset. The use of csv files to store extracted data resulted necessarily as data extraction is an extremely time-consuming process. In order to speed up further steps and experiments in the works each spectrogram was saved once for all and its data was used several times.

B. Further features elaboration

As presented in section II, in this work were proposed two different approaches to the urban sound classification problem. For each of these methods was used a different data elaborations accordingly to the neural network structure used.

In the first case was studied a CNN model to classify audio files. Stored Mel spectrograms were reshaped into 3D images and used as input of the neural network.

In the second proposed method, extracted data were furtherly elaborated.

For each audio frame, we extracted Mel-frequency capestral coefficients (MFCCs) using the standard librosa function and giving as input the pre-extracted mel-spectrogram.

Mel Frequency Cepstral Coefficients (MFCCs) are a powerful audio feature that can be generated by computing a discrete cosine transform on Mel spectrogram data. The transformed data are a more compact representation of the audio data than the Mel spectrogram and are commonly used in environmental sound analysis. MFCCs are often used as a benchmark in sound classification as shown also in [7] and in [5].

For each MFCCs frame (20 in total) extracted from an audio file, we recorded the following statistics: minimum, maximum, median, average, variance, skewness and kurtosis. The dimension of the overall obtained array has a length of 140. This array was used as input of the second method studied in this work.

Most of these features were extracted using Numpy [15] standard function, while skewness and kurtosis were computed using Scipy.stats specific functions [16].

C. Label representation

In both models labels were converted in One-Hot encoding representation.

One-Hot encoding is a representation of categorical data very common in machine learning. Some kinds of ML models can operate directly using label data directly. Neural Networks requires the output data to be numerical.

The label is represented as an array of size equals to the total number of classes. Each class correspond to one index of the array. All the elements are set to 0 except the element x_i where i is the index corresponding to the label to be represented.

For example, in this work the array:

$$[1.0.0.0.0.0.0.0.0.0.]$$

represent the air conditioner class, since the index associated to the label is 0. This representation allows standardization of classes representation.

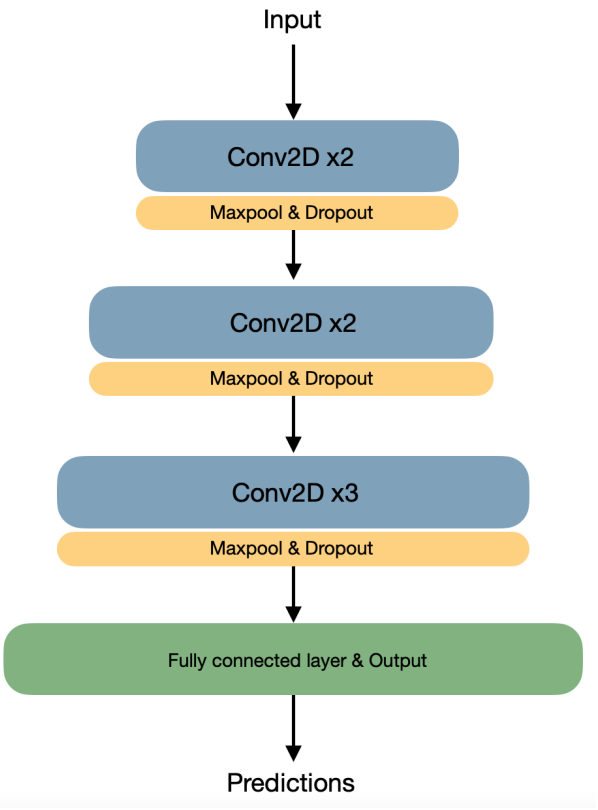


Figure 13. Architecture of the developed Convolutional Neural Network.

IV. Models Development

Building a Neural Network involves several decisions to be made about both architecture (number of inputs, number and dimension of layers and other size-related layer parameters) and hyperparameters (batch size, learning rate, dropout percentage). The selection process is still heuristic and there are no real rules to guide the research of best parameters. On the other side, an exhaustive evaluation of all the possible parameters in every possible combination is unfeasible.

Therefore the selection of the most promising model was done on a limited number of parameters and on quite small Neural Networks.

In this section are present the two approaches to the audio classification problem. The different model architectures and components are discussed in detail.

A. Convolutional Neural Network

The first model is a Convolutional Neural Network to which input data was passed as an image.

After the preprocessing and storing data phase described in Section (III-A) the mel-spectrogram of training data was resized to get a 3-D image of size 128x156x1 and used as Network input.

The Neural network structure is based on the following layers:

- The first two convolutional ReLu layers consisted of 32 filters of square shape (5x5 size, 1×1 stride). Max-pooling was applied with a pool shape of 3x3. Dropout was applied with a percentage of 40%.
- Following there are two ReLu layers consisting of 64 filters of shape 3x3. Max-pooling was applied with a pool shape of 2x2. Dropout was applied with a percentage of 40%.
- Further processing was applied adding three ReLu layers consisting of 128 filters of shape 2x2. Max-pooling was added with a pool shape of 2x2. Dropout was applied with a percentage of 40%.
- Finally was added one ReLu Fully connected layer and an output layer with Softmax Activation function.

In Figure 13 is depicted the general CNN structure.

During the training phase of the network was used the Categorical cross-entropy loss. This kind of loss is developed for multi-label classification tasks since it evaluates the loss for each class for a given sample independently. It calculates the distance between the discrete probability distributions predicted by the model and the true prediction. It computes the following sum:

$$L = -\sum_{i=1}^{K} y_i \cdot \log \hat{y}_i$$

where $\hat{y}_i$ is the i-th scalar value in the model output and y_i is the corresponding value of the true correct output. K is the number of classes.

The Categorical cross-entropy was chosen instead of the Binary cross-entropy (a similar but simpler function for multi-label classification) because it provides a faster convergence during the training phase.

Adam optimizer was used for stochastic gradient descent. Adam optimizer is an extension of the stochastic gradient descent algorithm and it is broadly adopted in deep learning.

During CNN development was clear the advantage of choosing a deeper model which had fewer parameters (smaller layers). This approach resulted less prone to overfitting. On the other side, more layers were added to the NN, more training epochs were required in the training phase to obtain optimal performances.

In the CNN the size of filters is quite small (5x5 in the first two layers, 3x3 and 2x2 in the following) as it showed higher performances than a network with filters of a bigger size. In literature there are several examples of performing CNN which uses bigger filters [8] [11], but in the studied case increasing the size of filters had no benefits.

The number of neurons in each layer was varied during the model development; experiments showed that increasing the number of neurons more we go deeper into the network improved the model performance.

More attempts were made to reduce the number of neurons per layer and adding more layers to the network at the same time but in this case model performance drop by 10%.

The dropout percentage was another studied hyperparameter. The Network gave the best result when the percentage was set at 40% for all the Dropout layers.

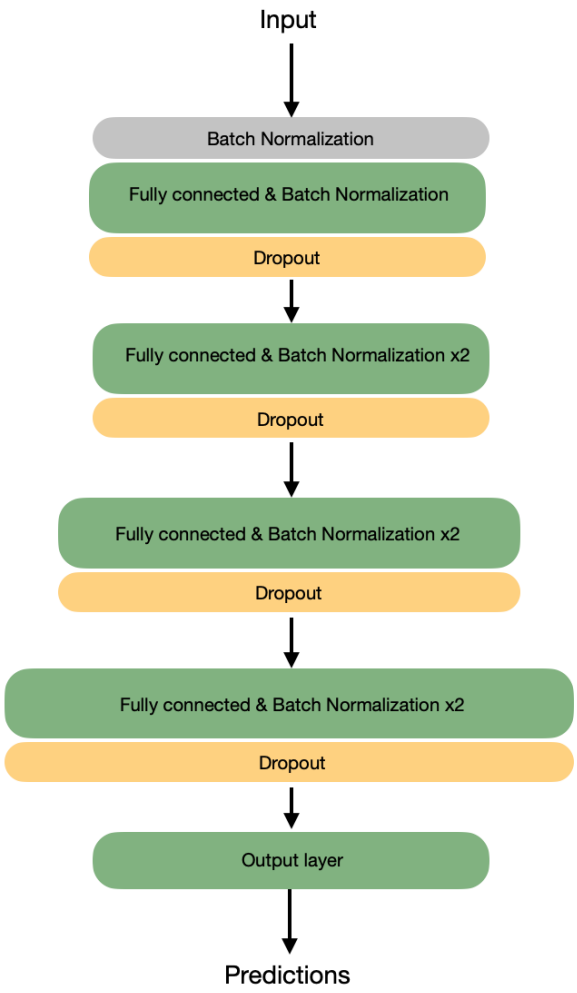


Figure 14. Architecture of the developed Feed forward Neural Network.

B. Feedforward neural network

The second model that was analysed was a Feedforward neural network. Its input was no more the whole Mel spectrogram as it was in the CNN, but only some selected features extracted from the Mel Frequency Cepstral Coefficients as described in section III-B.

Each input data has 1x140 dimensions, where 140 is the total number of MFCCs extracted features for each audio file.

The neural network has seven layers plus the output layer and it is structured as follow:

- A Batch Normalization input layer, followed by a Relu Fully connected layer consisting of 256 neurons, Batch Normalization and a Dropout layer with dropout rate of 40%.
- A ReLu Fully connected layer consisting of 256 neurons and Batch Normalization repeated twice, followed by Dropout layer with a dropout rate of 40%.
- A ReLu Fully connected layer consisting in 512 neurons and Batch Normalization repeated twice, followed by Dropout layer with adropout rate of 40%.
- A ReLu Fully connected layer consisting in 1024 neurons and Batch Normalization repeated twice, followed by Dropout layer with a dropout rate of 40%.
- Finally there is an output layer with a Softmax activation function.

In Figure 14 is depicted the general structure.

Just like in the CNN, in the training phase it was used Categorical cross-entropy as loss function and Adam optimizer in gradient descent algorithm.

Feedforward NNs have fewer parameters to train than previously presented CNN. In the phase of the model development, the network was mainly evaluated changing the number of layers and the number of neurons in each layer.

The introduction of Batch Normalization layers helped in reducing the whole model error standardizing layers input. Batch normalization applies a transformation to the data that shifts the mean output close to 0 and the output standard deviation close to 1. Batch Normalization is a method that accelerates the training phase, reducing the number of epochs.

Just like in the CNN the best performances were obtained when the number of neurons in each layer increased as the layer went deeper. In the first layer, the number of neurons is 256 and it doubles as we "move down" in each group of layers. Surprisingly, increasing the number of neurons in layers does not improve performances.

Each group of layers is followed by a Dropout layer with 40% of drop rate, which resulted to be the best solution.

C. Software tools

Both in CNN and Feedforward development, neural network architecture building, training and testing was pivotal the use of the Keras library [17]. It provides all the primitives for the building of a neural network. In data analysis and management was very useful Scikit-learn library [18] in addition with more standard python libraries such as Pandas [19], Numpy [15] and Mathplotlib [20].

D. Further observations

During models development were performed some attempts of building deeper neural networks than the presented ones (both in the case of the CNN and in Feed forward NN). The addition of further layers to models, even reducing layers size, caused immediate overfit of the models and performances dropped dramatically.

Low performances for deeper neural networks may be due also to the relatively small size of the dataset. The given dataset could in fact not be sufficiently big to train properly a neural network deeper than a certain threshold as the training could not be able to "reach" deeper layers of the model.

V. Results

During the CNN training was performed 10-fold cross-validation while in the feed forward Neural Network 5-fold validation provides better results. Each fold was trained for 20 epochs and was used a batch size of 32.

Several experiments were made to choose the batch size; 32 resulted to be the optimal solution between 16, 32, 62, 64 and 128 as batch size.

The number of training epochs for each folder was chosen observing the variation of the loss and validation loss. In the performed experiments, after the 20th training epoch training and validation losses only oscillates without any further convergence.

The final CNN and Feed forward models have respectively an average accuracy of 60.79% and 59.55% across test folds. Standard deviation was 2.2% and 3.6% for the CNN and the Feed forward NN respectively.

In Figure 15 is depicted a comparison between the two model performances according to accuracy, precision, recall and f1-scores criteria. CNN performances are sightly superior to the Feed forward neural network for all evaluated criteria, although the difference is minimal.

The weighted average for all classes of the f1-score of the CNN results is 60.8% and 59.3% for the Feed forward Neural network.

These performances are on the same line with several other results obtained in literature, such as the one obtained in [11], [12]. However the results of the quoted articles can't be significantly compared with the ones in this work. Most of these experiments are based on the use of a bigger training set than the one used in this work and therefore better-trained.

We can also analyse how the models perform for each category. Table I displays criterion of precision, recall and f1-scores for the CNN divided for each predicted urban sound class. Table II displays same informations for the feed forward neural network.

Although the average accuracy of the two models is not gratifyingly high; the average f1-score rate of the top 5 classes is about 74%. In the CNN the 5 best-recognised classes are car horn, dog bark, gun shot, siren and street music; the average f1-score across these is about 78%. In the Feed forward NN the average f1-scores of the top 5 classes(that in this case are car horn, dog bark, engine idling, gun shot and siren) is about 70%.

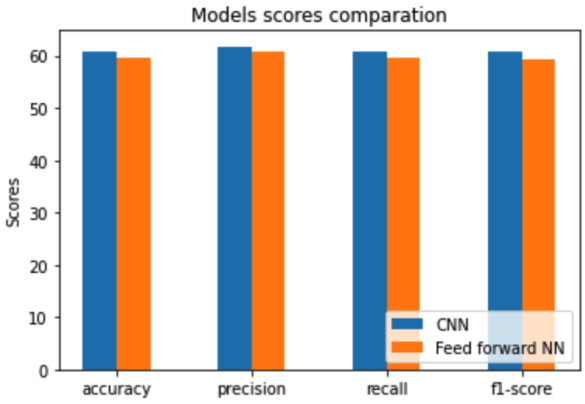


Figure 15. Comparison of the tho model perfomances on the basis of accuracy, precision, recall and f1-score metrics.

Class	precision	recall	f1-score
air conditioner	0.5497	0.4980	0.5226
car horn	0.7782	0.9050	0.8368
children playing	0.6606	0.6500	0.6552
dog bark	0.8440	0.7360	0.7863
drilling	0.4223	0.3640	0.3910
engine idling	0.4748	0.4679	0.4713
gun shot	0.7218	0.9728	0.8287
jackhammer	00.3676	0.4823	0.4172
siren	0.8900	0.6794	0.7706
street music	0.6527	0.7140	0.6819

Table I
CLASSIFICATION PERFORMANCES FOR THE CNN.

Class	precision	recall	f1-score
air conditioner	0.5745	0.5400	0.5567
car horn	0.6751	0.6018	0.6364
children playing	0.4973	0.7260	0.5902
dog bark	0.8154	0.6360	0.7146
drilling	0.4497	0.5180	0.4814
engine idling	0.7455	0.6004	0.6651
gun shot	0.7455	0.9076	0.8186
jackhammer	0.4585	0.3296	0.3835
siren	0.6942	0.7048	0.6995
street music	0.5460	0.5820	0.5634

Table II
CLASSIFICATION PERFORMANCES FOR THE FEED FORWARD NN.

Comparing data of Tables I and II we first can observe that the top 5 classes between the two models are quite completely overlapping. In particoular, the dog bark and gun shot classes have very high recognition rates in both models.

On the other side in both models jackhammer and drilling classes are recognised with a quite low recognition rate.

It is interesting to notice that this disparity of performances between these groups of classes has no connection to the dataset number of samples.

The number of samples of the gunshot sound is quite small in the training set as it consists of only 190 excerpts. However, gunshot class recognition rate is the highest among all the labels. This may denote that the used data representation (that is Mel spectrogram and MCCSs) is well suitable in describing gunshot sound specific features.

On the other side, the jackhammer class results well represented in the training set, as the number of excerpt recording this sound are 540.

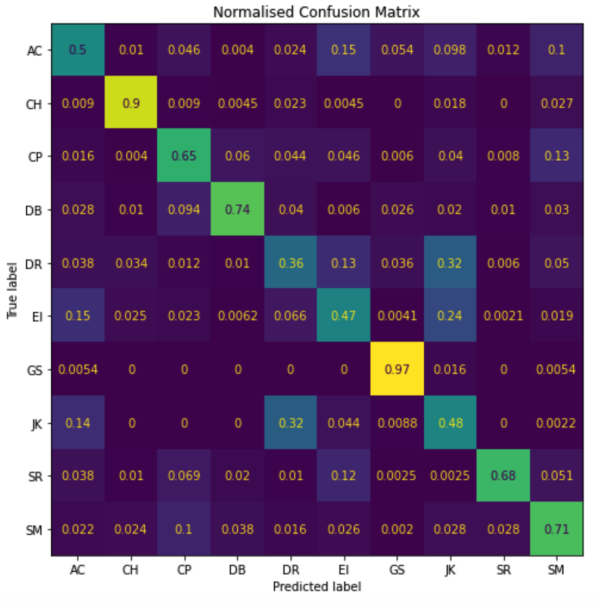
Despite the large representation of the jackhammer class in the training set its recognition performances are lower than any other class.

In a complementary way to what we said about the gunshot class, we may think that the Mel spectrogram representation and its use in the models do not allow to capture properly specific characheristics of the jackhammer sound.

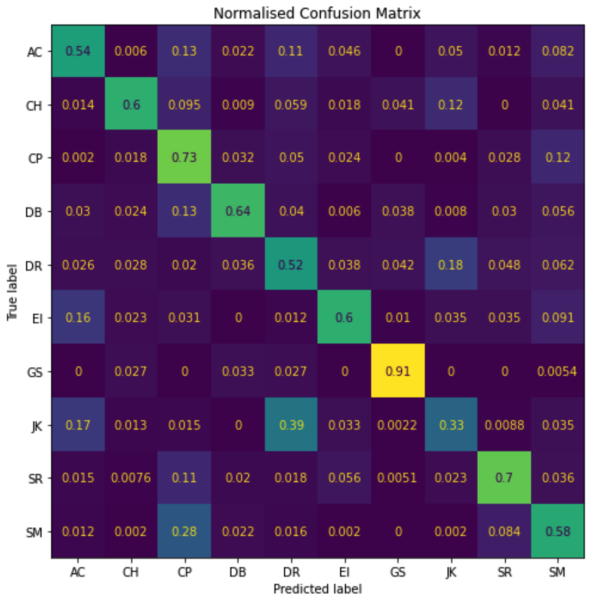
In Figures 16(a) and 16(b) are depicted the confusion matrix of the two developed models.

In both confusion matrices, the gunshot class is clearly distinguished from all other classes, while the jackhammer is quite often confused with other classes.

Confusion matrix, also called matrix error, is a specific table



(a) Confusion matrix for the CNN model.



(b) Confusion matrix for the Feed Forward model.

Figure 16. Classes: air conditioner (AC), car horn (CH), children playing (CP), dog bark (DB), drilling (DR), engine idling (EI), gun shot (GS), jackhammer (JK), siren (SR), street music (SM).

layout to represent error across classes. It is very common in machine learning, in particular in supervised learning tasks. In the graph, each row of the matrix represents the actual label of the class and each column represents the model prediction. Each cell contains the value with which each element belonging to the class of its row is labelled as the column class.

The CNN confusion matrix denotes a quite high confusion between jackhammer, drilling and engine idling classes. The drilling and jackhammer are quite commonly mismatched. It is probably due to the characteristic vibrating sound of both classes. In addition, engine idling is often labelled as jackhammer.

This data is not surprising when we compare it with our personal experience. To the human hearing, the sound of an engine idling and a jackhammer could be confused in the distance. The involvement of the drilling sound in the confusion is although more surprising.

It is interesting to notice that in the Feed forward NN, jackhammer audio excerpts are mostly confused only with drilling, without involving engine idling. This means that this model is more capable of distinguishing this kind of class than CNN.

The Feed forward confusion matrix denotes a quite high mismatch rate between street music class and children playing one. This is peculiar of this model as in the CNN equivalent graph the children playing class is recognised quite clearly. The confusion between street music and children playing classes has a strong influence over the children playing recognition rate that is significantly lower than in the CNN (30%).

VI. Conclusions

The goal of this work was to build two neural networks models capable of classifying urban sound records. The two models proposed were a classical Feedforward neural network that takes features extracted from MFCCs elaborated from audio files and a convolutional neural network that uses Melspectrogram images as inputs.

Although the accuracy performance of the two models was about of 60%, half of the classes were recognised with significant reliability. The top 5 recognised classes have an average f1-score of 78% and 70% respectively for the CNN and the Feedforward Neural Network.

In the performance comparison of the two models, the CNN approach resulted particularly interesting. The use of CNN is therefore promising not only in image recognition problems but also in sound classification tasks. Its performances were slignly better than the feed foreward neural networks, using a smaller number of layers.

However serious limitation of this work was the relatively small size of the data set. The quality and dimension of the pool of data used in training dramatically influence deep neural network models performances. The use of a bigger pool of data may allow a better training of the network, improving its final performances.

Since labelled data are costly and difficult to obtain, a possible solution to this problem could be the use of data augmentation.

Data augmentation techniques apply simple deformations to the audio sample and the deformed data are added to the pre-existing dataset; this would enlarge the data pool and provide new artificial but realistic data. By training neural networks with this enlarged pool, the model would be more capable of generalising rules and have better performances on unseen data.

A possible further work could be focused on the development of an augmented dataset. That might be used to train the two models presented in this work in order to compare the performances of models trained with the standard dataset and the enlarged one and finally evaluate the actual impact of the use of a bigger dataset over a smaller one.

REFERENCES

[1] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[2] H. Lee, P. Pham, Y. Largman, and A. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," *Advances in neural information processing systems*, vol. 22, pp. 1096–1104, 2009.

[3] M. McKinney and J. Breebaart, "Features for audio and music classification," 2003.

[4] H. Purwins, B. Li, T. Virtanen, J. Schlüter, S.-Y. Chang, and T. Sainath, "Deep learning for audio signal processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 2, pp. 206–219, 2019.

[5] J. B. Martin F. McKinney, "Features for audio and music classification," 2003.

[6] J. P. B. Justin Salamon, Christopher Jacoby, "A dataset and taxonomy for urban sound research," *22nd ACM International Conference on Multimedia, Orlando USA, Nov*, 2014.

[7] M. S. K. C. Keunwoo Choi, Gyo rgy Fazekas, "A comparison of audio signal preprocessing methods for deep neural networks on music tagging," *26th European Signal Processing Conference (EUSIPCO)*, 2018.

[8] I. S. A. Krizhevsky and G. E. Hinton, "Ima- genet classification with deep convolutional neural net- works," *Advances in Neural Information Processing Systems*, 2012.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[10] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data mining and knowledge discovery*, vol. 33, no. 4, pp. 917–963, 2019.

[11] K. J. Piczak, "Environmental sound classification with convolutional neural networks." *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP). IEEE*, 2015.

[12] T. S. Zhang B., Leitner J., "Audio recognition using mel spectrograms and convolution neural networks," 2019.

[13] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal processing letters*, vol. 24, no. 3, pp. 279–283, 2017.

[14] Librosa, https://librosa.org/.

[15] Numpy, https://numpy.org/doc/stable/index.html.

[16] Scipy.stats, https://docs.scipy.org/doc/scipy/reference/stats.html.

[17] Keras, https://keras.io/.

[18] Scikit-learn, https://scikit-learn.org/stable/.

[19] Pandas, https://pandas.pydata.org/.

[20] Matplotlib, https://matplotlib.org/.

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.