SOUND EVENT CLASSIFICATION

Chiara Auriemma, Francesca Benesso, Anna Fusari, Filippo Marri

Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano
Piazza Leonardo Da Vinci 32, 20122 Milano, Italy
[chiara.auriemma,francesca1.benesso]@mail.polimi.it
[anna.fusari,filippo.marri]@mail.polimi.it

ABSTRACT

Sound Event Classification (SEC) has become an important task in the field of audio processing, with applications ranging from environmental monitoring to human-computer interaction. Aim of this project is to develop a sound event classification system based on a Convolutional Neural Network (CNN) architecture training it on the ESC-50 dataset. The model proposed is a multi-branch convolutional neural network (MBCNN) architecture that processes the log mel-spectrogram of the audio signal. It achieves an average test accuracy of 86%. At the end of the paper, the performance of the model is compared with some state-of-the-art models.

Index Terms— Sound Event Classification, Multi-branch Convolutional Neural Network, ESC-50 dataset

1. INTRODUCTION

1.1. Background

Sound Event Classification (SEC) is a task that consists in classify specific sound events within an audio signal. This task has gained significant attention in recent years due to its wide range of applications, including environmental monitoring [1], human-computer interaction [2], and multimedia content analysis [3]. The goal of SEC is to accurately classify sound events in real-time or from pre-recorded audio data. The process of SEC typically involves several steps, including feature extraction, model training, and evaluation [4].

Commonly used features include log-mel spectrograms, Mel-frequency cepstral coefficients (MFCCs) and log-spectrograms. Model training involves using labeled audio data to train a machine learning model to classify sound events. Various machine learning algorithms can be used for SEC, including support vector machines (SVMs), decision trees, and deep learning models such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) [5]. The algorithm choice depends on the computational power available and on the amount of data used.

Evaluation of the SEC system is typically done using standard metrics such as accuracy, precision, recall, and F1-score.

1.2. State-of-the-art

As happened in most of the research fileds, the advent of neural networks in the context of sound event classification have significantly improved the task. Traditional approaches, which typically involved feature engineering and classical machine learning algorithms, such as Support Vector Machines (SVM), k-Nearest Neighbors (kNN) and Multi-Layer Perceptrons (MLP), have been surpassed by modern deep learning architectures such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs). These architectures have demonstrated superior performance in capturing the intricate and complex patterns present in audio signals. Aside that, techniques like log-Mel spectrogram extraction and data augmentation have helped improve model generalization. Furthermore, emerging hybrid models that integrate CNNs with attention mechanisms or transformer-based architectures have shown promise in enhancing the modeling of temporal and spatial features in sound data. Recent advancements also include transfer learning approaches using pre-trained CNN architectures. These methods not only enhance accuracy but also address challenges related to training time [6, 4]. We list now some interesting models we used as reference for our project.

1.2.1. Piczack model

One of the first attempts in using 2D-CNN for SEC was made by Piczak [7] in 2015, in which the log mel-spectrogram is split in several segments and each segment is processed individually using a CNN. At the end, a series of Dense Layer is used to classify the segments. Final predictions for a sound are generated using either a majority-voting scheme or by taking into account the probabilities predicted for each segment. Piczak's model is notable for being a banchmark model of this task.

1.2.2. Attention based model

In 2020, Zhang et al. [8] proposed an attention-based convolutional recurrent neural network (ACRNN) for SEC. The model combines convolutional layers for feature extraction with recurrent layers to capture temporal dependencies in the audio signal. The attention mechanism allows the model to focus on specific parts of the audio signal that are more relevant for classification, improving performance on complex sound events. A remarkable point of this paper is the kind of input used: instead of having classical log mel-spectrograms, the authors used a gammatone spectrogram that leads to better results in terms of classification accuracy.

1.2.3. CRNN model

Another model we used as reference to develop our architecture is the Convolutional Recurrent Neural Network (CRNN) proposed by Kiran Sanjeevan Cabeza [9]. There, a hybrid model is implemented: firstly three 2D convolutional layers are used to extract features from the log mel-spectrogram, then two recurrent layers

are used to capture temporal dependencies. Finally, a softmax layer is used to classify the sound events.

1.2.4. End-to-end model

An interesting approach we found in literature is the one adopted for the model written by Sang et. al. [10]. There, both feature extraction and classification are performed by the neural network that takes as input directly the raw data. An advantage of a model like this one is surely the one of being more accessible. In fact, it is not necessary needed deep knowledge reguardind the task since everything is done by the model itself.

1.2.5. GoogLeNet model

The key innovation of GoogLeNet [11] is the Inception module, which allows the network to capture features at multiple scales simultaneously by combining convolutions of different sizes within the same layer. This design significantly reduces the number of parameters compared to traditional deep networks, making it more efficient without sacrificing performance. In the article by Boddapati et al. [12], is shown how the GoogLeNet architecture can be adapted to audio classification task using transfer learning.

1.2.6. DenseNet201 model

The DenseNet201 [13] is a 201 layers newtork that finds the right trade-off between the learning of very complex features and efficiency. It has been shown to achieve excellent performance on image classification benchmarks, such as ImageNet, by enhancing information propagation and reducing overfitting. The article considered as a reference for the implementation of this model for audio is the one by Ashurov [6] in which several models, among them DenseNet201, are applied to audio tasks according to the transfer learning technique.

2. METHODOLOGY

As a baseline of this project we chose a simple 2D Convolutional Neural Network (CNN) architecture based on the Conv2D model of laboratory number 4. We made this choice since we have, as input, a log mel-spectrogram. In turn, this kind of input has been chosen in order to feed the network with temporal-frequency correalated data, approach widely followed in the literature. However, the results obtained with this architecture were not satisfactory, so we decided to improve the model by adding some layers and using a more complex architecture. After several attempts with Recurrent Convolutional Neural Networks (CRNN), we opted for a multi-branch Convolutional Neural Network (MBCNN) architecture inspired by the work of Enes Furkan Örnek [14, 15].

2.1. Feature extraction

As hinted before, the neural network is fed with a log mel-spectrogram, computed by a specific function that takes as input the signal preserving its original sample rate. The parameters are the following:

- **N_fft**: 1024 samples
- **Hop size**: 512 samples
- **Number of mel bands**: 128 (predefined parameter)

The output is a 2D array of shape (128, 1723) representing how energy in different Mel frequency bands evolves over time.

2.2. Model description

The Jupyter Notebook for this model can be found at the following link:

`https://github.com/ChiaraAuriemma/Sound-Event-Classification`.

Our implementation utilizes a bi-dimensional Convolutional Neural Network (CNN) with four parallel branches, each extracting different levels of information from the input data using specifically calibrated kernel sizes. We highlight that our model is working in a peculiar way with the 2D layers since one dimension is, layer-by-layer, alternatively imposed equal to 1. With this method, each branch alternates between horizontal and vertical filter orientations, enabling efficient extraction of localized patterns across both time and frequency domains. The net is built following a cellular approach. Each cell is composed by a Conv2D layer followed by batch normalization and ReLU activation in order to ensure training stability and non-linearity. The representation of a single cell is reported in Fig 1.

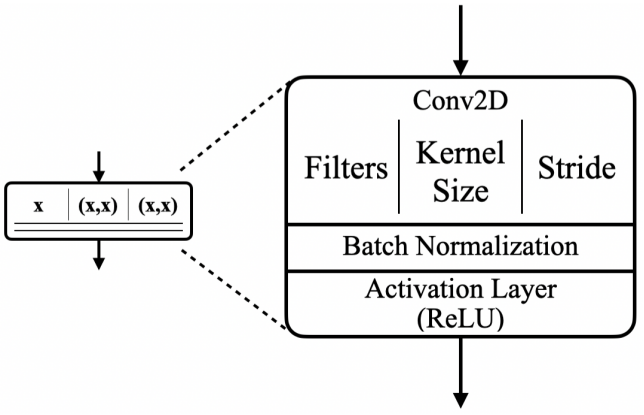

Figure 1: Structure of a cell of the architecture.

After processing through these branches, the outputs are merged via element-wise addition, followed by additional convolutional layers with a higher number of filters to further abstract the combined feature maps. A global average pooling layer condenses the spatial information, and the network concludes with a dense softmax layer that outputs class probabilities over 50 categories. The entire architecture is depicted in Fig. 2.

The model is optimized using the Adam optimizer with AMSGrad and learning rate equal to 0.00005, and trained using sparse categorical cross-entropy loss.

The theoretical justification for our filter size selection derives from signal processing principles in audio analysis. Smaller filters (1×8) effectively capture localized patterns and high-frequency components, while larger filters (1×64) extract broader patterns and low-frequency information.

Not only mathematical reasons lie behind our approach, but also biological ones. In facts, the human auditory system employs a wave bank filter mechanism to capture specific frequency components in audio signals, with different regions of the basilar membrane responding to distinct frequency ranges. By incorporating

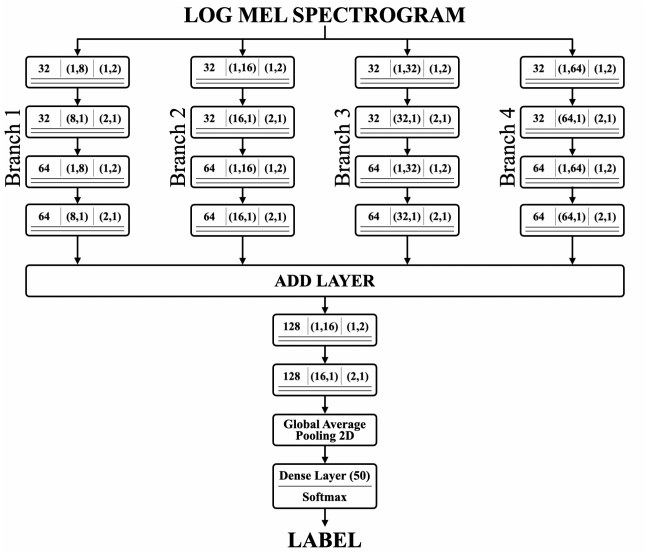

Figure 2: Architecture of the multi-branch convolutional neural network.

parallel branches with carefully selected filter sizes, the network similarly detects patterns within specific frequency ranges, enabling multiscale feature representation.

3. EVALUATION

3.1. Dataset analysis and preprocessing

The dataset used for this project is the well-known ESC-50 dataset [16], which contains 2000 labeled isolated eviromental sound events from 50 different classes, with each class containing 40 samples. Each sound of the dataset is a mono recording available in WAV format (Ogg Vorbis compress at 192 kbit/s) with a sample rate of 44.1 kHz and a bit depth of 16 bits. Clips in this dataset have been manually extracted from public field recordings gathered by the Freesound project [17]. The resulting dataset is available under a Creative Commons non-commercial license through the Harvard Dataverse project [16].

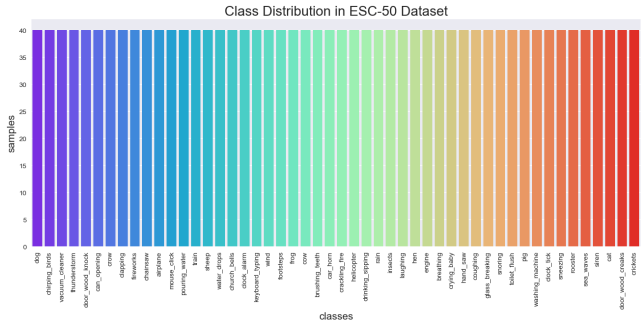

Figure 3: Graphical demontration of the balance of the dataset used.

According to the analysis that will be done on the results inspired by *Attention based Convolutional Recurrent Neural Network for Environmental Sound Classification* [8], the type of sound events in the dataset can be divided into three main categories:

- **Transient sounds**: this category includes sounds that have a short duration and are characterized by a sudden onset, such as the one of a glass breaking, a thunderstorm, or a firework.
- **Continous sounds**: this category includes sounds that have a longer duration and are characterized by a continuous or sustained sound, such as the one a vacuum cleaner, a car engine running, or pouring water.
- **Intermittent sounds**: this category includes sounds that have a periodic or irregular pattern, such as the one of a dog barking, a bird chirping, or a man coughing.

Some examples are reported in Fig. 4, where we can see the power spectrogram of a transient sound (glass breaking), a continuous sound (vacuum cleaner), and an intermittent sound (dog barking).

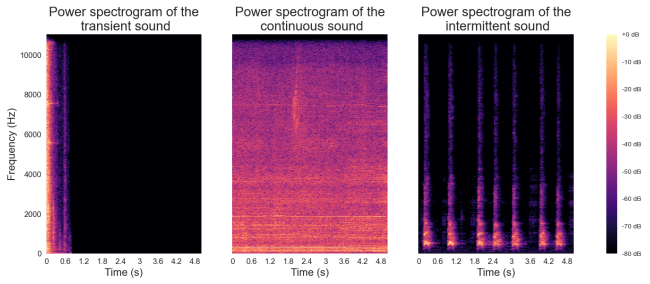

Figure 4: Power spectrogram of a transient, a continous and an intermittent sound.

According to what is reported on the paper in which the ESC-50 dataset is presented [16], we higlight how some sounds are more difficult to classify than others, such as the sounds of a washing machine, an helicopter, or an engine due to their similar spectrograms. If we look to Fig. 5, we can see how the three spectrograms are extremely similar.

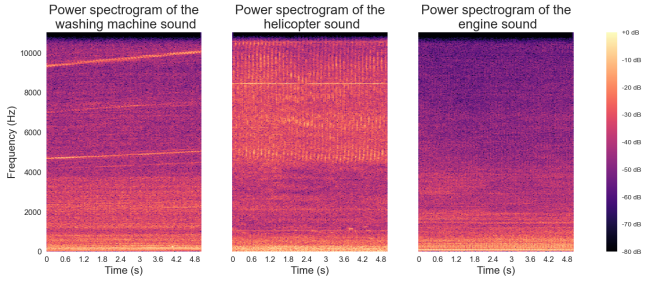

Figure 5: Power spectrogram of the sound produced by a washing machine, an helicopter and an engine.

This misclassification happens not only for machines, but for humans too. This will be taken into account in the results section, where we will see how the model performs on different classes of sounds.

It is also important to observe that, even though we consider the same class, the variability of the sounds is very high, as we can see by comparing the spectrograms of three different samples of the *dog barking* class. As we can see in Fig. 6, the three spectrograms

are very different one from each other. According to the classification defined at the beginning, the first one can be considered an impulsive sound, the second one a continuous sound, and the third one an intermittent sound. This means that, given the nature of the sound in this class, the onset information is not so useful.

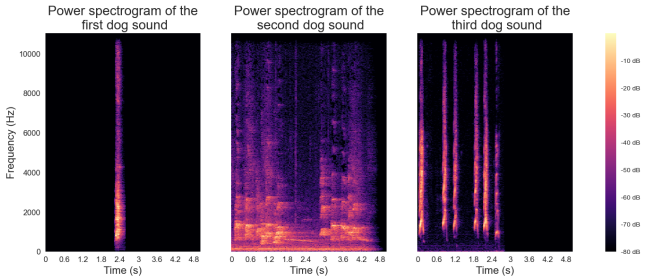

Figure 6: Power spectrogram of three different sounds belonging to the same *dog barking* class.

Furthermore, we underline how some of the ambiental sounds, like the one of the wind, have no characteristic structure: by breaking down their spectrograms in their harmonic and percussive components, as it is done in Fig. 6, it is evident that the difference it is not so clear since the two plots are almost equal.

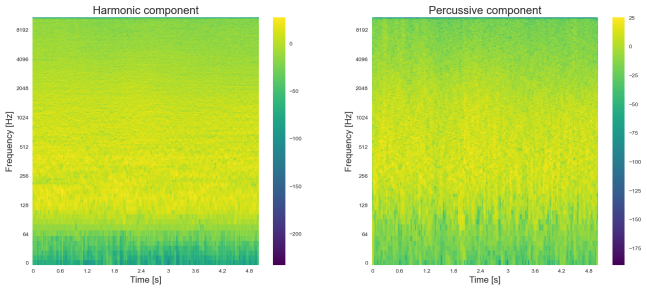

Figure 7: Harmonic and percussive decomposition of the wind blowing sound.

This fast analysis of the dataset has been done to understand the limitations of the model and the difficulties that it could encounter during the training phase.

A part of the dataset, 200 elements (10% of the total), has been separated for testing. The remaining samples have been splitted according to the `stratifiedkfold` module of sklearn [18] in a training and a validation set.

Drawing inspiration by the Salamon and Bello paper [19], five different techniques have been implemented in order to perform data augmentation on the training set:

- **Time Stretching (TS)**: the audio signal is stretched or compressed in time by a random factor within a specified range. A last boolean parameter active the cropping of the processed audio to the original length, so that the model can be trained on the same length of the original audio.
- **Pitch Shifting (PS)**: the audio signal is shifted in pitch by a random factor within a specified range.
- **Background Noise (BN)**: a Gaussian noise is added to the audio signal with a specified SNR range.
- **Dynamic Range Compression (DRC)**: the dynamic range of the audio signal is compressed from a certain threshold with a specified ratio, attack time, and release time. This function is implemented using the `Compressor` module of the library `pedalboard` by Spotify [20].
- **Convolution with Impulse Responses (CIR)**: the audio signal is convolved with the *MIT Acoustical Reverberation Scene Statistics Survey* dataset of impulse response [21] to simulate different acoustic environments.

For all the results presented in this paper, the training dataset has been preprocessed using the following parameters:

- TS: factor between 0.8 and 1.25
- PS: factor between -5 and 5 semitones
- BN: SNR between 5 and 40 dB
- DRC: threshold of -20 dB, ratio of 4:1, attack time of 10 ms, release time of 100 ms
- CIR: active

At the end of the preprocessing phase, the training set has been augmented by a factor of 6 (original + 5 augmented clips), so that the model can be trained on a larger dataset.

3.2. Evaluation metrics

The evaluation of the sound event classification system is performed using several metrics to assess its performance. Firstly, the test accuracy and the loss for the training and validation sets. Then, the classification reports with the others metrics (precision, recall and F1-score) and the confusion matrix are computed to evaluate the overall performance of the model.

To train our model, we use 5 folds cross-validation, which allows us to obtain a more robust estimate of the model's performance. Furthermore, we ensured that there is no data leakage between training and validation sets by doing data augmentation only on the training set after the splitting.

The training of the models have been performed on the A100 GPU by Google Colab.

4. RESULTS

4.1. Main model results

The results of the training of the model for each fold are reported in Fig. 8, 9, 10, 11 and 12.

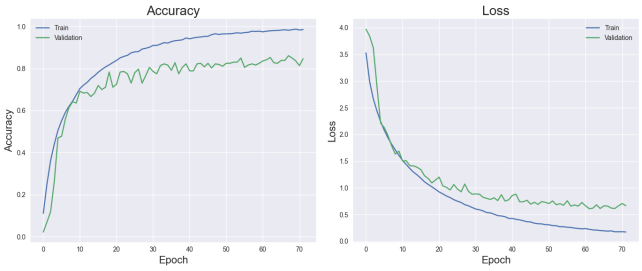

Figure 8: Result of the training for the zero fold.

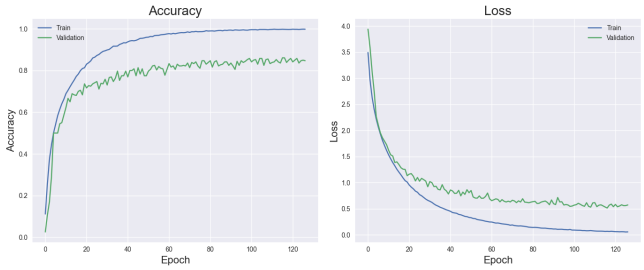

Figure 9: Result of the training for the first fold.

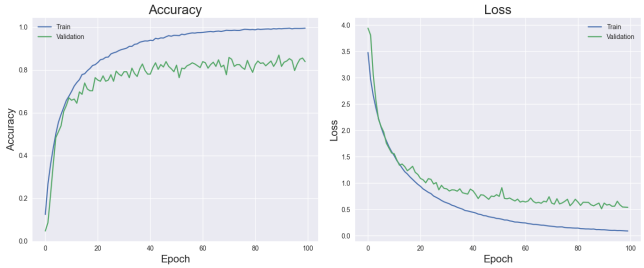

Figure 10: Result of the training for the second fold.

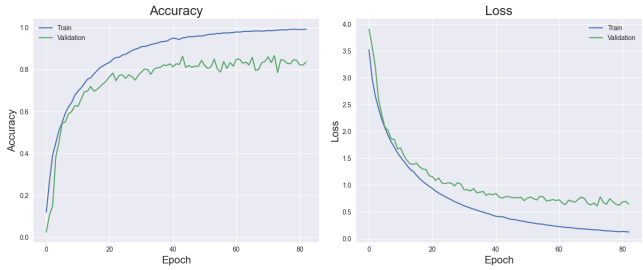

Figure 11: Result of the training for the third fold.

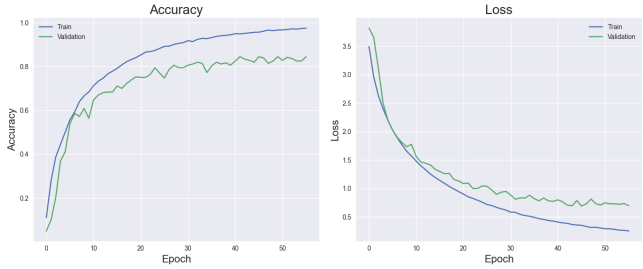

Figure 12: Result of the training for the fourth fold.

The average test accuracy over the 5 folds reached by the model is 86% with an average loss of 0.63. The performance for each fold is reported in Table 1.

Table 1: Results of the main model for each fold.

Fold	Test Accuracy	Test Loss
0	88%	0.58
1	87%	0.59
2	88%	0.54
3	82%	0.70
4	83%	0.75
Average	86%	0.63
Standard Deviation	2.4%	

In Fig. 13 and Fig. 14 the validation and the test confusion matrices of the model with the best accuracy (88%) are reported. The labeling of the classes is reported in the appendix (chapter 7).

The classification reports are located in the Appendix (chapter 7) too where Table 5 is refered to the validation and Table 6 is refered to the test.

We highlight here how precision and recall are well balanced meaning that the model is not biased towards a specific class.
By looking at the test confusion matrix, we can see that the model is able to classify correctly most of the sounds, with some exceptions. One of them is the *wind* class that is misclassified 3 times out of 4: one time classified as *train*, one as *airplane* and as *sea waves*. As we can notice, all the three sounds are continuous and noisy sounds, so it is not surprising that the model has some difficulties in classifying them. This match with what we assumed by looking at Fig. 5: sounds with similar spectrograms are more difficult to classify. It is interesting to notice that the recall of these three classes is not high even in the results of the experiment conducted by Piczak [16], in which a group of humnas listened to the sounds and classified them. This leads us to think that, since the model is inspired by the human auditory system, it can have its same difficulties in classifying the sounds.

Another class that is misclassified is the *coughing* class, which is confused with the *sneezing* and *water drops* class. We underline how all these sounds are intermittent sounds and how they have similar spectrograms.

For what it concerns false positive, these are the errors made by the model: the sound of the *airplane* and *helicopter* has been classified as *wind*, the sound of *sneezing* and *laughing* as *caughing* and, for two times, the *glass breaking* sound has been misclassified as *can opening*. We can justify these results by claiming the same motivation we exposed before: similar spectrograms lead to misclassification.

By looking at the classification report of the baseline model (the 3 2D-CNN-layer model of lab 4 trained on a non augmented dataset), we notice that these same classes have been misclassified. However, we cannot directly infer that there are some biases toward them, since the test accuracy of the baseline model is pretty low (53%), leading us to think that the model is not strong enough even to perform a proper misclassification: it is always not so able to understand which sounds are similar.

With a slightly stronger version of the baseline (a 6 2D-CNN-layer model trained on a augmented dataset), the accuracy reaches 72% (with still a high loss) returning a more reliable result for classification report comparison. There, we can see that, this time again, classes with noisy and similar spectrograms have been misclassified. For instance, in validation, as shown in Fig. 15, the *wind* sound has classified 2 times out of 8 as an *airplane* sound, result that we got for our MBCNN model too. For the test, as reported in Fig. 16, we got a miscalssification of the same *wind* sound, that have been classified 1 time out of 4 as a *vacuum cleaner* sound. It is also interesting to analyze the behavior of the model with the ghost classes: this time again, we are dealing with classes with noisy spectrograms as the one cited in Fig. 5. In fact, we see that the sounds belonging

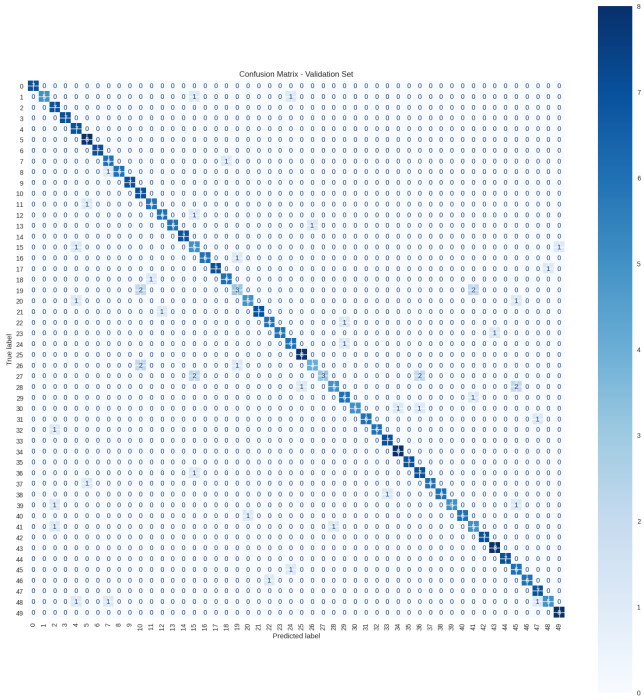

Figure 13: Confusion Matrix of the validation set for our MBCNN model.

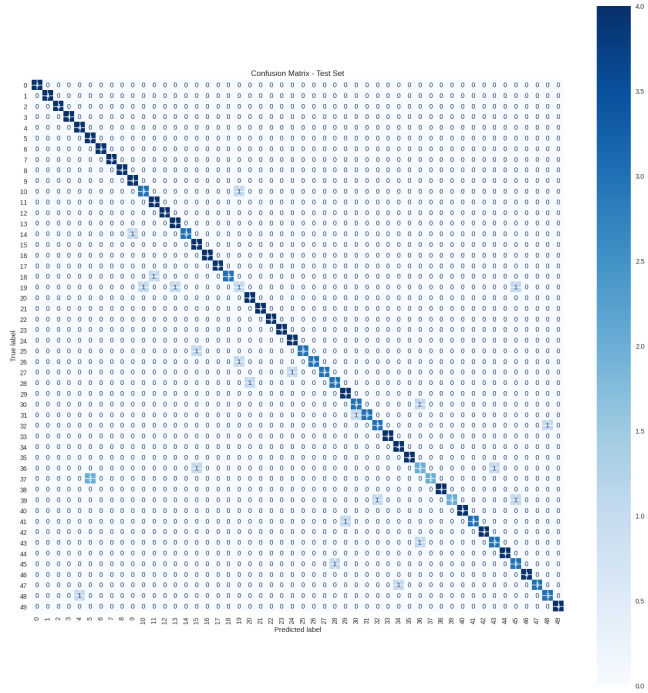

Figure 14: Confusion Matrix of the test set for our MBCNN model.

to the ghost class *engine* have been classified as sounds belonging to the *train* sound class, the *helicopter* sound class, the *coughing* sound class and the *washing machine* sound class. The same we can notice for the sound belongin to the ghost class *washing machine* that have been classified as sounds belonging to the *thunderstorm* sound class, the *train* sound class, the *clock alarm* sound class and the *engine* sound class.

4.2. Comparison: MBCNN model with different inputs

During the training of our MBCNN model, we tried to use different inputs to see how they affect its performance. At the end, we noticed that all of them shows good results, with a slight improvement in the accuracy when using the log mel-spectrogram as input. They are reported in Table 2.

Table 2: Performances for different inputs.

Input	Test Accuracy	Test Loss
log mel-spectrogram	86%	0.63
log STFT	83%	0.71
MFCCs	78%	0.78

We notice how MFCCs are the worst input. This could seems strange thinking that MFFCs are the most processed input we can use among the ones tested. MFCCs could be effective with classical machine learning algorithms, where the features were extracted from the raw data and then used to train the model. However, with the improvements made by neural networks, it is no longer necessary to perform classical feature engineering since the networks are able to extract better features by themselves. For what we just said, we can undesrtand why even the simple STFT works better than MFCCs.

4.3. Comparison: MBCNN model with others

We report here some results of the comparison between our model and others: in Table 3 comparisons between different models implemented by us are reported, while in Table 4 comparisons with other models from the literature are reported.

Table 3: Performances of different models implemented by us.

Model	Dataset	Test Accuracy	Test Loss
MBCNN	ESC-50	86%	0.63
Baseline	ESC-50	53%	3.12
6 2D-CNN-layer	ESC-50	72%	2.78
Our attention based	ESC-50	81%	1.61
Our CRNN	ESC-50	60%	1.8

From now on, with the prefix *our*, we refer to models inspired by literature but implemented by us from scratch. The main difference between the original models and ours are related to the data augmentation techniques used, as for *our* models we used the same data augmentation techniques described in Section 3.1. This justifies the inconcistencies between the results of our models, shown in Table 3 and the ones of the literature presented in Table 4.

Due to the limitations of our computational resources, we used a simplified version of the pre-processing phase for what it concerns the *our* attention based model. However, the results achieved by our test are similar to the exposed in the literature, as we can see in Table 4.

At a glance, performances seem different for *our* CRNN model

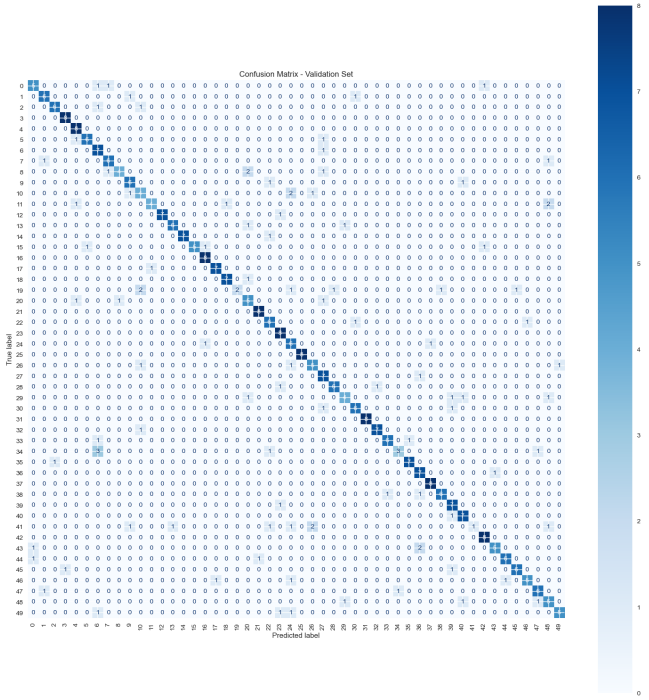

Figure 15: Confusion Matrix of the validation set for the 6 2D-CNN-layer model.

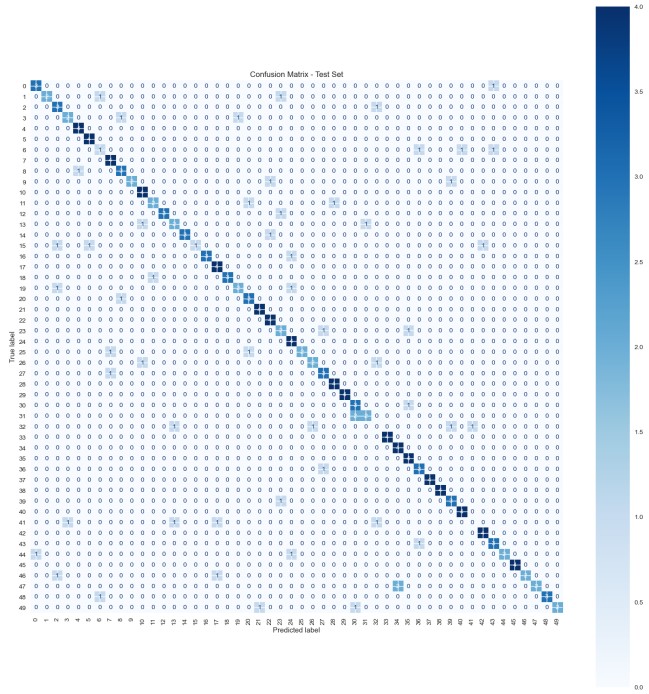

Figure 16: Confusion Matrix of the test set for the 6 2D-CNN-layer model.

Table 4: Performances of different models of the literature.

Model	Dataset	Test Accuracy
Piczak [7]	ESC-50	64.5%
Piczak [7]	UrbanSound8K	73.1%
Attention Based [8]	ESC-50	86.1%
CRNN [9]	UrbanSound8K	73.5%
End-to-end [10]	UrbanSound8K	79%
GoogLeNet [12]	ESC-50	73%
GoogLeNet [12]	UrbanSound8K	93%
DenseNet201 [6]	UrbanSound8K	97.25%

with respect to the Cabeza's one [9]. However, we underline that the dataset to which the two models are trained is different: we used the ESC-50 dataset, while Cabeza's model is trained on the UrbanSound8K dataset. This leads us to think that this architecture is not so effective for the ESC-50 dataset.

Our personal attempts on hybrid architecture (like CRNN) have not been so successful. Since simpler convolutional models (number of parameters of CRNN is around 4 million, while the one of *our* MBCNN is around 1 million) are able to reach better results, we decided to focus on them until we reached the final model.

At the end, the results of our final model appears to be competitive with the ones we saw in the literature. The only model that outperform *our* MBCNN version is the DenseNet201 [6], model based on transfer learning. However it has been trained on the UrbanSound8K dataset, which containes less classes and less ambiguous sounds like the ones we saw in Section 3.1, so we cannot assure that the performance are similar if trained and tested on ESC-50.

5. ACKNOWLEDGMENTS

This work was supported by the Politecnico di Milano, within the framework of the Selected Topics in Music and Acoustic Engineering Course 2025.

6. REFERENCES

[1] R. Narasimhan, X. Z. Fern, and R. Raich, "Simultaneous segmentation and classification of bird song using cnn," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 146–150.

[2] S. Cunningham, H. Ridley, J. Weinel, and R. Picking, "Supervised machine learning for audio emotion recognition," *Personal and Ubiquitous Computing*, vol. 25, no. 4, pp. 637–650, 2021. [Online]. Available: https://doi.org/10.1007/s00779-020-01389-0

[3] A. Kumar and B. Raj, "Weakly supervised scalable audio content analysis," 2016. [Online]. Available: https://arxiv.org/abs/1606.03664

[4] S. Padmaja and N. Sharmila Banu, "A systematic literature review on sound event detection and classification," in *2025 5th International Conference on Trends in Material Science and Inventive Materials (ICTMIM)*, 2025, pp. 1580–1587.

[5] A. Bansal and N. K. Garg, "Environmental sound classification: A descriptive review of the literature," *Intelligent Systems with Applications*, vol. 16, p. 200115, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2667305322000539

[6] A. Ashurov, Y. Zhou, L. Shi, Y. Zhao, and H. Liu, "Environmental sound classification based on transfer-learning techniques with multiple optimizers," *Electronics*, vol. 11, no. 15, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/15/2279

[7] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015, pp. 1–6.

[8] Z. Zhang, S. Xu, T. Qiao, S. Zhang, and S. Cao, "Attention based convolutional recurrent neural network for environmental sound classification," 2019. [Online]. Available: https://arxiv.org/abs/1907.02230

[9] K. S. Cabeza, "Pytorch audio classification: Urban sounds," https://github.com/ksanjeevan/crnn-audio-classification, 2019, accessed: 2025-06-10.

[10] J. Sang, S. Park, and J. Lee, "Convolutional recurrent neural networks for urban sound classification using raw waveforms," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 2444–2448.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014. [Online]. Available: https://arxiv.org/abs/1409.4842

[12] V. Boddapati, A. Petef, J. Rasmusson, and L. Lundberg, "Classifying environmental sounds using image recognition networks," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 356–360.

[13] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, 2016. [Online]. Available: http://arxiv.org/abs/1608.06993

[14] E. F. Örnek, "Audio classification using cnn on esc-50 dataset," https://github.com/sweat0198/audio_classification_CNN_ESC-50, 2023, accessed: 2025-06-09.

[15] S. A. Latifi, H. Ghassemian, and M. Imani, "Classification of heart sounds using multi-branch deep convolutional network and lstm-cnn," 2025. [Online]. Available: https://arxiv.org/abs/2407.10689

[16] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *Proceedings of the 23rd Annual ACM Conference on Multimedia*. ACM, 2015, pp. 1015–1018.

[17] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "Fsd50k: An open dataset of human-labeled sound events," *arXiv preprint arXiv:2010.00475*, 2020, accessed: 2025-06-09. [Online]. Available: https://arxiv.org/abs/2010.00475

[18] S. learn developers, "Stratifiedkfold — scikit-learn 1.5.2 documentation," 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html

[19] 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.

[20] P. Sobot and Spotify, "Pedalboard: A python library for real-time audio effects," July 2023. [Online]. Available: https://github.com/spotify/pedalboard

[21] J. Traer and J. H. McDermott, "Statistics of natural reverberation enable perceptual separation of sound and space," *Proceedings of the National Academy of Sciences*, vol. 113, no. 48, pp. E7856–E7865, 2016. [Online]. Available: https://www.pnas.org/doi/10.1073/pnas.1612524113

7. APPENDIX

Sound classes of the ESC-50 dataset.

- 0: dog
- 1: chirping birds
- 2: vacuum cleaner
- 3: thunderstorm
- 4: door wood knock
- 5: can opening
- 6: crow
- 7: clapping
- 8: fireworks
- 9: chainsaw
- 10: airplane
- 11: mouse click
- 12: pouring water
- 13: train
- 14: sheep
- 15: water drops
- 16: church bells
- 17: clock alarm
- 18: keyboard typing
- 19: wind
- 20: footsteps
- 21: frog
- 22: cow
- 23: brushing teeth
- 24: car horn
- 25: crackling fire
- 26: helicopter
- 27: drinking sipping
- 28: rain
- 29: insects
- 30: laughing
- 31: hen
- 32: engine
- 33: breathing
- 34: crying baby
- 35: hand saw
- 36: coughing
- 37: glass breaking
- 38: snoring
- 39: toilet flush
- 40: pig
- 41: washing machine
- 42: clock tick
- 43: sneezing
- 44: rooster
- 45: sea waves
- 46: siren
- 47: cat
- 48: door wood creaks
- 49: crickets

Table 5: Classification report on the validation set related to the model with best test accuracy (fold 2).

Class	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	7
1	1.00	0.71	0.83	7
2	0.70	1.00	0.82	7
3	1.00	1.00	1.00	7
4	0.70	1.00	0.82	7
5	0.80	1.00	0.89	8
6	1.00	1.00	1.00	7
7	0.75	0.86	0.80	7
8	1.00	0.86	0.92	7
9	1.00	1.00	1.00	7
10	0.64	1.00	0.78	7
11	0.86	0.86	0.86	7
12	0.86	0.86	0.86	7
13	1.00	0.86	0.92	7
14	1.00	1.00	1.00	7
15	0.50	0.71	0.59	7
16	1.00	0.86	0.92	7
17	1.00	0.88	0.93	8
18	0.86	0.86	0.86	7
19	0.60	0.43	0.50	7
20	0.83	0.71	0.77	7
21	1.00	0.88	0.93	8
22	0.86	0.86	0.86	7
23	1.00	0.86	0.92	7
24	0.75	0.86	0.80	7
25	0.89	1.00	0.94	8
26	0.80	0.57	0.67	7
27	1.00	0.43	0.60	7
28	0.83	0.62	0.71	8
29	0.75	0.86	0.80	7
30	1.00	0.71	0.83	7
31	1.00	0.86	0.92	7
32	1.00	0.86	0.92	7
33	0.88	1.00	0.93	7
34	0.89	1.00	0.94	8
35	1.00	1.00	1.00	7
36	0.70	0.88	0.78	8
37	1.00	0.86	0.92	7
38	1.00	0.86	0.92	7
39	1.00	0.71	0.83	7
40	1.00	0.86	0.92	7
41	0.62	0.71	0.67	7
42	1.00	1.00	1.00	7
43	0.89	1.00	0.94	8
44	1.00	1.00	1.00	7
45	0.60	0.86	0.71	7
46	1.00	0.86	0.92	7
47	0.78	1.00	0.88	7
48	0.83	0.62	0.71	8
49	0.89	1.00	0.94	8
Accuracy			0.86	360
Marco avg	0.88	0.86	0.86	360
Weighted avg	0.88	0.86	0.86	360

Table 6: Classification report on the test set related to the model with best test accuracy (fold 2).

Class	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	4
1	1.00	1.00	1.00	4
2	1.00	1.00	1.00	4
3	1.00	1.00	1.00	4
4	0.80	1.00	0.89	4
5	0.67	1.00	0.80	4
6	1.00	1.00	1.00	4
7	1.00	1.00	1.00	4
8	1.00	1.00	1.00	4
9	0.80	1.00	0.89	4
10	0.75	0.75	0.75	4
11	0.80	1.00	0.89	4
12	1.00	1.00	1.00	4
13	0.80	1.00	0.89	4
14	1.00	0.75	0.86	4
15	0.67	1.00	0.80	4
16	1.00	1.00	1.00	4
17	1.00	1.00	1.00	4
18	1.00	0.75	0.86	4
19	0.33	0.25	0.29	4
20	0.80	1.00	0.89	4
21	1.00	1.00	1.00	4
22	1.00	1.00	1.00	4
23	1.00	1.00	1.00	4
24	0.80	1.00	0.89	4
25	1.00	0.75	0.86	4
26	1.00	0.75	0.86	4
27	1.00	0.75	0.86	4
28	0.75	0.75	0.75	4
29	0.80	1.00	0.89	4
30	0.75	0.75	0.75	4
31	1.00	0.75	0.86	4
32	0.75	0.75	0.75	4
33	1.00	1.00	1.00	4
34	0.80	1.00	0.89	4
35	1.00	1.00	1.00	4
36	0.50	0.50	0.50	4
37	1.00	0.50	0.67	4
38	1.00	1.00	1.00	4
39	1.00	0.50	0.67	4
40	1.00	1.00	1.00	4
41	1.00	0.75	0.86	4
42	1.00	1.00	1.00	4
43	0.75	0.75	0.75	4
44	1.00	1.00	1.00	4
45	0.60	0.75	0.67	4
46	1.00	1.00	1.00	4
47	1.00	0.75	0.86	4
48	0.75	0.75	0.75	4
49	1.00	1.00	1.00	4
Accuracy			0.88	200
Macro avg	0.89	0.88	0.88	200
Weighted avg	0.89	0.88	0.88	200