Data Augmentation for Environmental Sound Classification - How to Deal with Small Datasets

Federico Chiarello†

***Abstract*—This paper evaluates the potential of audio-based data augmentation in classifying short audio clips of environmental sounds. The datasets used in this study (ESC-10) pose significant challenges due to the limited number of samples. To takle this issue different audio augmentation strategies have been implemented. The classification task has performed using convolutional neural networks trained on mel spectrograms.**

***Index Terms*—Environmental Sound Classification, Data Augmentation, Convolutional Neural Networks.**

I. Related Work

A pivotal point in the environmental sound classification research area has been represented by the public release of benchmark datasets, such as the ones used in this study (ESC-50 and ESC-10) [1] or the UrbanSound8K dataset [2]. The ESC datasets offer a well-curated collection of environmental sound recordings with ESC-50 comprising 2,000 audio clips organized into 50 balanced classes, and its subset, ESC-10, focusing on 10 particularly representative categories providing researchers with balanced and diverse sound examples. In contrast, the UrbanSound8K dataset gathers 8,732 urban audio excerpts across 10 classes, capturing the complexity and variability of real-world city sounds. Building on these benchmarks, numerous approaches have been explored and compared while trying to enhance classification performances.

Piczak's work [3] demonstrated that using mel spectrogram representations as input to CNNs effectively captures the time frequency characteristics of environmental sounds. He extracted log mel-spectrograms form the audio samples and computed their deltas (first temporal derivative). Then he obtained a two dimensional feature map by stacking the log mel-spectrogram and the delta. This two-channel input resembles the classical structure of RGB images, commonly used to train Convolutional Neural Networks.

An alternative approach is represented by Ensamble Stacked Convolutional Neural Networks [4], in which mel spectrograms are used as input features alongside raw audio data. This dual representation strategy aims to exploit complementary information where mel spectrograms provide a condensed frequency view and raw waveforms preserve finer temporal details thereby potentially offering more robust feature extraction and improved recognition performance.

Given the clear temporal components of audio data, an alternative and intuitive approach compared to CNNs, consists in training Recurrent Neural Networks. A clear benefit of those models is represented by their inherent ability to ingest input of variable size. This has been explored in the context of Unsupervised Representation Learning from audio data [5]. A recurrent sequence to sequence autoencoder has been trained for unsupervised representation learning from mel-spectrograms extracted from the raw audio files. Then the learnt representations have been extracted from the fully connected layer placed between the decoder and encoder units and have been used to train a multilayer neural network for the original classification task.

†Data Science, Department of Mathematics, University of Padova, email: federico.chiarello.1@studenti.unipd.it, ID: 2058163

II. Processing Pipeline

A data preparation pipeline has been developed in order to efficiently build a dataset to be used to train different Convolutional Neural Networks architectures. The raw audio samples were loaded and used to compute their respective Log Mel Spectrogram. Than a two dimensional feature map was obtained by stacking the log mel-spectrograms and their deltas (first temporal derivatives). Mel spectrograms were choose as the main building block of the input feature vectors due to their ability to effectively capture the spectral and temporal properties of audio signals.

Two different CNN architectures were tested. The first one will be referred as Baseline and the second one as MelNet. The MelNet architecture was inspired by an architecture proposed in [4], with the addition of Dropout layers for regularization.

Test were conducted on the effectiveness of Audio Augmentation techniques and on their impact on the model performances. A TensorFlow Dataset pipeline was used to generate the different versions of the datasets (differentiated by the presence or absence of data augmentation). In particular the TensorFlow Dataset pipeline covered the following points:

- Creating the dataset from file names and labels;
- Mapping a custom function to load the raw audio samples;
- Caching the dataset composed by raw audio samples to avoid redundant computation;
- Shuffling the data;
- Repeating the dataset indefinitely;
- Applying random audio augmentations (if required);
- Computing mel-spectrograms and deltas and stacking them;
- Normalizing the spectrograms and deltas;
- Batching the data;
- Prefetching batches to improve performances.

The results obtained by the different experiments were tested on an apposit test set.

III. Signals and Features

A. ESC-10 Dataset

The ESC-50 dataset consists of 2000 5-second-long recordings organized into 50 semantical classes (with 40 examples per class) loosely arranged into 5 major categories and it is designed for benchmarking environmental sound classification methods. The ESC-10 dataset (subset of ESC-50) is used in this project. It is organized into 10 classes from the original 50.

Those classes are:

- dog
- rooster
- rain
- sea waves
- crackling fire
- crying baby
- sneezing
- clock tick
- helicopter
- chainsaw

B. Log-Mel Spectrograms

Log-scaled mel-spectrograms were extracted from all recordings (resampled to 22050 Hz) with window size of 1024, hop length of 512 and 60 mel-bands. The Log-scaled mel-spectrograms delta (first temporal derivative) was then computed. Finally a two dimensional feature map is obtained by stacking the log mel-spectrogram and the delta.

Mel spectrograms are commonly used in speech processing and machine learning tasks, they show the frequency content of audio signals over time. It is common practice to express the strength of the mel frequency components in decibels. This is commonly referred to as a log-mel spectrogram. The mel scale is a perceptual scale that approximates the non-linear frequency response of the human ear.

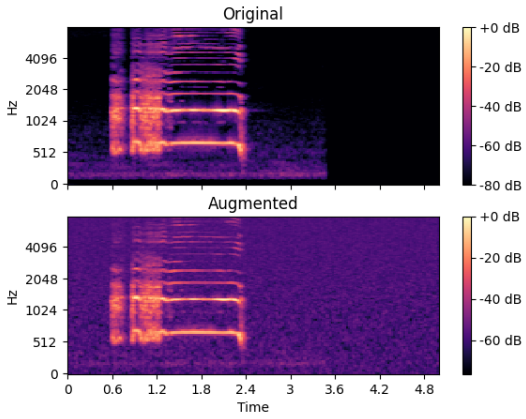

Fig. 1: Log Mel-Spectrogram

C. Audio Augmentations

A series of Audio Augmentation techniques have been implemented:

- Time Stretching: Randomly speed up or slow down the audio by a certain rate, cropping or padding the audio with Gaussian noise scaled by a noise factor;
- Pitch Shifting: Shifts the pitch of the audio by up to a specified number of semitones in either direction;
- Gaussian Noise: Add Gaussian noise to the audio signal;
- Audio Shifting: shifts the audio in time by a random amount, up to a fraction of the audio's length. Fills the empty region with Gaussian noise scaled by a noise factor.

Two different Augmentation Strategies (A1 and A2) where designed. Each one of the different transformations were applied to the audio sample with a probability of 0.5 in the A1 schema and with probability 0.6 in the A2 schema. Audio Shifting was not applied if the sample has already been transformed using Time Stretching, since it would be redundant. The idea behind the creation of two different augmentation schemas was to test the effectiveness of those transformations and to access whether they help improving the models performances. The two schemas differ also for the parameter used by the transformations reported in the following table.

Audio Augmentations	Schema A1	Schema A2
Time Stretching	(0.75, 1.25)	(0.5, 1.5)
Pitch Shifting	2	4
Gaussian Noise	0.01	0.02
Max Shifting	0.2	0.4
Padding	0.002	0.005

TABLE 1: Audio Augmentation Parameters for Schema A1 and Schema A2

D. Dataset Splitting

The dataset was already arranged into 5 uniformly sized cross-validation folds, ensuring that clips originating from the same initial source file are always contained in a single fold. Due to the limited computational resources availability I decided to not perform cross validation, and instead train my models on the first 3 folds, using fold number 4 as a validation set for hyper-parameter tuning and the fold 5 as test set for the final models evaluation and comparison. This decision was motivated by the limitation imposed by the computational resources at my disposal.

IV. Learning Framework

A. Baseline

The Baseline network was implemented following the standard guidelines for CNN architectures design. The architecture is structured as follows:

- The network employs four sequential convolutional layers with increasing filter counts (32, 64, 128, and 256)

and decreasing kernel sizes (6, 5, 4, and 3 respectively). Each convolutional layer is immediately followed by a max-pooling layer (with pool size 2), which downsamples the feature maps and helps in capturing the most salient features.
- Transition to Fully Connected Layers: After the convolutional stages, the feature maps are flattened into a one-dimensional vector. A dropout layer with a 50% rate is applied to reduce overfitting by preventing co-adaptation of neurons.
- The flattened output is then passed through two dense layers with 200 and 100 neurons respectively, both using the ReLU activation function to learn non-linear combinations of the features.
- The final dense layer applies a softmax activation function to produce probability distributions over the target classes.

The model is compiled with the Adam optimizer and uses sparse categorical crossentropy as its loss function. This model was designed as a starting point for this project experiments

B. MelNet

The MelNet architecture was inspired by an architecture proposed in [4], with the addition of Dropout layers for regularization. The architecture is structured as follows:

- Two convolutional layers with 24 filters and a kernel size of 6 are applied consecutively. A 20% dropout is introduced after the first convolution to help mitigate overfitting. A max pooling layer follows to reduce the spatial dimensions.
- The network increases the complexity with two convolutional layers using 48 filters and a kernel size of 5, with a stride of 2. Again, a 20% dropout is applied between these layers. This block is also followed by max pooling for further dimensionality reduction.
- A deeper convolutional layer with 64 filters and a kernel size of 4 (with a stride of 2) is used to capture more abstract features. This is immediately followed by batch normalization, which stabilizes and accelerates training.
- The extracted features are flattened and then passed through a dense layer with 200 neurons using ReLU activation. L2 regularization is applied to this layer to control overfitting, and a 50% dropout is used both before and after the dense layer to further enhance generalization.
- Finally, a softmax layer outputs the class probabilities for multi-class classification.

The model is compiled using the Adam optimizer and the sparse categorical crossentropy loss function. The main difference form the architecture that inspired this design is represented by the addition of Dropout layers for regularization. Those were needed since after some preliminary experiments the network showed a strong overfitting behavior. One peculiarity of this architecture is the absence of Max-Pooling Layers, motivated by the idea that their absence will allow the network to retain more information.

C. Training

Both models were trained from 100 epochs, with an early-stopping approach to prevent overfitting (stopping training if there was no improvement on the validation loss), and batch size 80. This setting was repeated for all the different experiments.

V. Results

The different results obtained by training the models using the two augmentation schemas and the training set without augmentations are reported in the following table. Those resuts were computed on the test set.

Model	Data Augmentation	Test Accuracy
Baseline	None	61.2
Baseline	A1	**66.3**
Baseline	A2	61.2
MelNet Reg	None	71.2
MelNet Reg	A1	68.8
MelNet Reg	A2	**75.0**

TABLE 2: Test Accuracy with Different Data Augmentation Strategies

We can clearly see that the MelNet architecture performed better in the classification task compared to the Baseline. The second augmentation schema (A2) proved to be the most effective with the MelNet architecture, suggesting that the higher variability introduced in the training set helped the model fitting procedure.

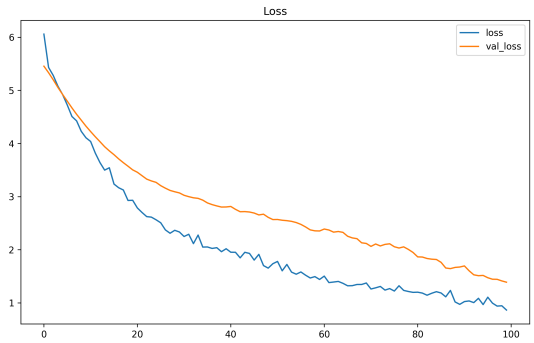

Fig. 2: MelNet with A2 Data Augmentation - Training Loss

The confusion matrix can give us an idea of which categories where misclassified by the model. As we can clearly see most of the times the model ends up misclassifing samples from categories that share some similarities.

VI. Concluding Remarks

In this project I tested the effectiveness of audio augmentation techniques in the context of Enviromental Sound Classification. A data preparation pipeline has been developed in order to efficiently build a dataset to be used to train different Convolutional Neural Networks architectures. Different audio augmentation techniques have been adopted

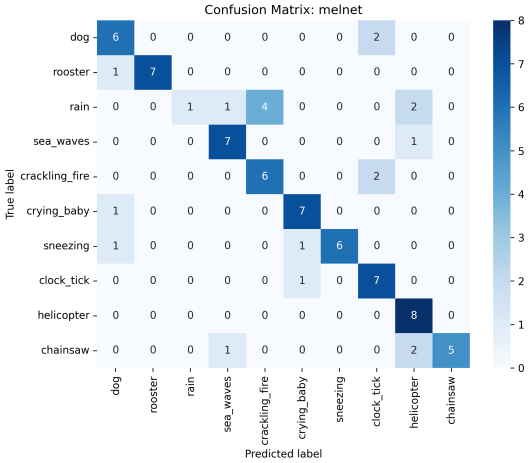

Fig. 3: MelNet with A2 Data Augmentation - Confusion Matrix

on raw audio samples, from which have then been extracted Log Mel Spectrograms, used as building blocks of the input feature vectors provided to the models.

The experiments confirmed that data augmentation could represent an interesting approach when dealing with audio dataset with low dimensionality.

Different challenges have been encountered during the development of this project. One challenging aspect of the dataset was his low dimensionality, which represent an unusual aspect in the data hungry domain of deep learning. Overfitting behavior clearly emerged from early prototypes of the models, requiring the adoption of different regularization techniques. Finally the main limiting factor in the exploration of different architectures or solutions has been the limited computational resources availability, which forced some less-than-ideal choices while designing the experiments.

REFERENCES

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

[2] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 1041–1044, 2014.

[3] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th international workshop on machine learning for signal processing (MLSP)*, pp. 1–6, IEEE, 2015.

[4] S. Li, Y. Yao, J. Hu, G. Liu, X. Yao, and J. Hu, "An ensemble stacked convolutional neural network model for environmental event sound recognition," *Applied Sciences*, vol. 8, no. 7, p. 1152, 2018.

[5] S. Amiriparian, M. Freitag, N. Cummins, and B. Schuller, "Sequence to sequence autoencoders for unsupervised representation learning from audio," in *DCASE*, pp. 17–21, 2017.