Comparison of different input features in CNN architectures for Environmental Sound Recognition

Mattia Varagnolo[†]

***Abstract*—Convolutional neural networks (CNNs) have been widely used for Sound recognition. In this study, we will see 3 different CNN models with multiple convolutional layers of decreasing filter sizes to improve the performance of CNN models using either log-mel feature input or raw waveform input. Additionally, we also benchmarked our model against one based on Mel-frequency cepstral coefficients (MFCCs). The benchmark is made on the famous ESC-50 dataset and the results highlights the effectiveness of the MFCCs for environmental sound recognition.**

***Index Terms*—Multi-Class Classification, Supervised Learning, Optimization, Neural Networks, Convolutional Neural Networks.**

I. Introduction

Environmental sound classification plays a crucial role in various applications such as surveillance, wildlife monitoring, and urban soundscape analysis. In recent years, Convolutional Neural Networks (CNNs) have emerged as powerful tools for sound event recognition due to their ability to automatically learn hierarchical features from audio data. Building upon the work of Shaobo Li [1], this project aims to deploy the architecture proposed in that paper and compare it with a model that use Mel-frequency cepstral coefficients (MFCCs), in order to assess performance differences in environmental sound classification task.

Our approach involves the development of two different CNN architectures: the first CNN utilizes log-mel features, while the second CNN processes raw waveform inputs. To enhance the performance, it has been proposed a five-layer stacked CNN network with a convolutional filter configuration featuring decreasing filter sizes. This configuration, combined with static and delta log-mel input features, aims to extract discriminative features for sound event recognition.

Furthermore, we present an end-to-end stacked CNN model designed specifically for sound event recognition from raw waveforms, eliminating the need for manual feature engineering. This model incorporates a two-layer feature extraction convolution layer and a specialized convolutional filter configuration to directly learn features from raw waveforms.

In this project, we evaluate the performance of this proposed models using the ESC-50 dataset, a widely used benchmark dataset for environmental sound classification. By conducting comprehensive experiments and comparisons, we aim to show the effectiveness of the different approaches, trying to improve sound event recognition accuracy.

II. Related Work

In the field of environmental event sound recognition (ESC), several key papers have contributed significantly to the understanding and advancement of the topic. One notable work by Davis and Mermelstein [2] compared parametric representations for monosyllabic word recognition in continuously spoken sentences, laying the groundwork for speech recognition research. Abdel-Hamid et al. [2] introduced Convolutional Neural Networks (CNNs) for speech recognition, showcasing the potential of deep learning in audio processing tasks. Jaitly and Hinton [2] explored learning better representations of speech soundwaves using restricted Boltzmann machines, further advancing the field of speech processing.

The study by Li et al. [3] presented a special CNN architecture for ESC, focusing on the MelNet and RawNet models. Their work analyzed the influence of different CNN architectures on recognition accuracy, setting a benchmark for future research in the field. Li et al. [3] tested CNN architectures with varying numbers of convolution layers and filter sizes, providing insights into the optimal network structures for ESC tasks.

Notably, our project diverges from the approach taken in Li et al. [3] by not incorporating the Dempster-Shafer (DS) evidence theory in our models. Instead, we focus on developing distinct architectures for ESC using raw waveform and log-mel feature inputs, without the utilization of DS theory. By conducting a comparative analysis, we aim to enhance the understanding of the effectiveness of different input features in ESC tasks.

III. Processing Pipeline

The processing pipeline is designed in the following way. We designed a series of functions to handle the data in input to be properly adapted for the different architectures. To train the MelNet, the input goes through the proper function, which elaborate the log-mel spectrogram with the delta, then we take just 1s and it goes through the architecture. The neural network is then trained for a number of epochs and the model is evaluated using a majority voting rule. This means that before to evaluate a sound, we give 8 segments in input and then we do the evaluation. The exact same procedure is done even for the other two models, the RawNet and the MFCCs Model. Based on the type of input needed for the architecture to be trained or evaluated, the audio signals are properly elaborated.

The loss function used is the ”Sparse Categorical Crossentropy”. Sparse Categorical Crossentropy is a loss function used

in classification tasks where the labels are provided as integers. It measures the difference between the true labels and the predicted probabilities.

Mathematically, it can be defined as:

$$J(w) = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i)$$

where:

- N is the number of samples.
- y_i is the true label (integer-encoded).
- $\hat{y}_i$ is the predicted probability for the true label.

Due to the fact that we are performing a multi-class classification task, this loss function efficiently handles integer labels.

Regarding the training details, we used adam as an optimizer but we implemented a dynamic learning rate just as it was proposed on the original paper of LI. The dynamic learning rate is designed as it follows:

$$learning\_rate = min\_learning\_rate + [max\_learning\_rate - min\_learning\_rate] \times e^{\frac{-iteration}{decay\_speed}}$$

We tried different batch sizes from 30 to 300 but the best results were with a value of 200 and it has been used for each model, even if the differences were slight. Regarding the epochs, we trained MelNet and Rawnet for 100 epochs, while MFCC network was trained for 300 epochs. More epochs appear to lead to overfitting so we found these values as the best compromise.

IV. Signals and Features

As we already said, we used three different kind of audio feature. The log spectrogram, with the delta, the raw waveform and the MFCCs. For the Log-Mel we used the librosa Python library to extract the log-scaled mel-spectrogram features with 60 bands to cover the frequency range (0 - 22,050 Hz) of the sound segments. At the same time, the sound segments are divided into 41 frames with an overlap of 50%, with each frame being about 23ms. Through these steps, we can represent the static log-scaled mel-spectrogram feature on each segment as a 60x40x1 matrix corresponding to frequency, time, channel. In addition, we calculate the first temporal derivative of log-mel on each frame to obtain the delta log-scaled mel-spectrogram feature, which is used as the second channel of input. Finally, the dimension of the extracted log-scaled mel-spectrogram feature maps is 60x41x2. We convert all sound files to monaural wav files with a sampling rate of 22,050 Hz. The length of each sound segment is 20,480 (about 1s) with a 50% overlap. We reshape the raw waveform segment from a one-dimensional form (20,480) into a two-dimensional matrix (1, 20,480) with one channel so that it matches the commonly used 2D filters in Tensorflow. Then we used the Mel-Frequency Cepstral Coefficients (MFCCs) for the MFCC model. MFCCs are a representation of the short-term power spectrum of a sound signal. They are commonly used in speech and audio processing because they capture the perceptually relevant aspects of the sound. The MFCCs are computed using the librosa library, which transforms the normalized audio signal into a set of 13 coefficients, resulting in a matrix where each column corresponds to the MFCCs of a frame in the audio. Then we append the energy of each frame to the MFCC matrix. Energy is a measure of the amplitude of the audio signal, providing additional information about the audio's intensity over time.

Subsequently, the MFCC matrix, along with its first and second-order differences (known as delta and delta-delta features), are converted into a TensorFlow tensor. These delta features capture the rate of change and acceleration of the MFCCs, respectively, and are calculated using the librosa.feature.delta function.

Finally, the tensor is transposed to shape the features appropriately for model input. The transposition permutes the dimensions of the tensor, ensuring that the feature dimension is in the correct position for subsequent processing steps in the pipeline, which in our case were 14x216x3

When we train the network, we randomly select these segments from the original training audio and input them into the prediction models. At each epoch, we choose different segments of the audio, but use the same training label, regardless of the selected segments. In the test phase, we input multiple segments of the audio file into the prediction network and perform a majority voting of the output prediction results for classification. We did not remove the silent section from the whole 5-s sound to preserve the integrity of the original audio. For the evaluation, though, we just consider the non-silent sections.

We split the train, test and validation set with a ratio of 0.7, having in total 1400 examples for the train set, 300 examples for the test set and 300 examples for the validation set. One important thing to say is that we didn't used a K-fold-cross validation scheme, because it was computationally hard and time consuming.

V. Learning Framework

The feature learning module in RawNet (Figure 1a) is designed to extract features from raw waveform inputs using convolutional layers akin to a bandpass filter bank. The module consists of two convolutional layers preceding a pooling layer, aimed at generating a two-dimensional feature vector similar to components extracted from a log-scaled mel-spectrogram.

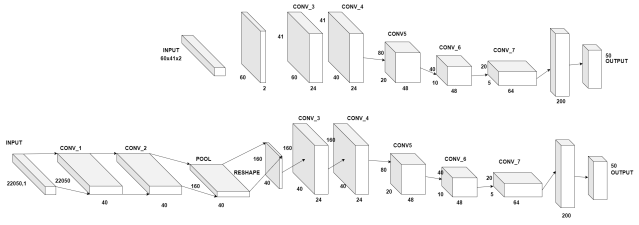

Fig. 1: Architecture of MelNet (Top) and Rawnet (bottom)

Each convolutional layer employs 40 filters with a receptive field of (1,8) and a stride of (1,1) in the time domain. This setup allows the model to capture local features across different time scales effectively, inspired by the success of EnvNet [5]. The input shape and parameters for these layers are summarized in Table 1.

Following the convolutional layers, a non-overlapping max pooling operation with a pooling size of 128 is applied. This operation reduces the spatial dimensions while preserving essential features. The resulting output matrix has dimensions (1, 160, 40), representing frequency, time, and channel dimensions. This matrix is reshaped to (40, 160, 1) to facilitate subsequent convolutional operations in both frequency and time dimensions.

The reshaped three-dimensional matrix (40, 160, 1) is then fed into the convolutional layer 'conv3', which performs classification tasks based on the learned time-frequency representations.

TABLE 1: PARAMETERS OF THE FEATURE LEARNING MODULE IN RAWNET

Layer	Input Shape	Filter	Kernel Size	Stride
Conv1	[batch, 1, 20480, 1]	40	(1, 8)	(1, 1)
Conv2	[batch, 1, 20480, 40]	40	(1, 8)	(1, 1)
Pool	[batch, 1, 20480, 40]	-	(1, 128)	-

The parameters specify the input shapes, number of filters, kernel sizes, and strides for each layer in the feature learning module of RawNet. The recognition modules of RawNet and MelNet share a common architecture:

- **Convolutional Layers:**
  - **L1-L5:** These layers consist of convolutional operations designed to capture hierarchical features from the input data. They employ varying numbers of filters and receptive fields:
    - **L1 & L2:** Each uses 24 filters with a (6×6) receptive field and stride (1×1), activated by ReLU.
    - **L3:** Utilizes 48 filters with a (5×5) receptive field and stride (2×2), followed by ReLU activation.
    - **L4:** Shares the parameters of L3: 48 filters, (5×5) receptive field, stride (2×2), and ReLU activation.
    - **L5:** Uses 64 filters with a (4×4) receptive field and stride (2×2), activated by ReLU.
- **Fully Connected Layers:**
  - **L6:** A fully connected layer with 200 hidden units and ReLU activation. This layer integrates high-level features from the convolutional layers.
  - **L7:** The output layer consists of either 10 or 50 units, followed by a softmax activation function, depending on the specific application.
- **Normalization and Regularization:**
  - Both models utilize Batch Normalization after convolutional layers to stabilize and accelerate the training process. Following Batch Normalization, a fully connected layer is employed.
  - In RawNet, a dropout probability of 0.5 is applied to the fully connected layer to mitigate overfitting and enhance generalization.

This architecture diverges from traditional CNN configurations by omitting max-pooling layers after convolution, aiming to retain more spatial information. Instead, Batch Normalization is utilized to enhance training efficiency and model robustness.

Regarding the MFCCs:

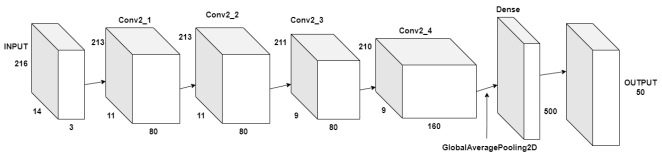

Fig. 2: Architecture of the MFCCs network

The proposed model architecture consists of multiple convolutional layers followed by batch normalization and dropout layers, structured to optimize feature extraction and classification performance. The input tensor is initially processed by a Conv2D layer, which applies 80 filters of size 4×4 with a stride of 1×1, using 'valid' padding. The ReLU activation function is employed to introduce non-linearity. Following this, batch normalization is applied to stabilize and accelerate the training process. A dropout layer with a dropout rate of 0.5 is then used to prevent overfitting by randomly setting a fraction of the input units to zero during training.

The output from this layer is passed into a second Conv2D layer, which again applies 80 filters, this time of size 3×3 with a stride of 1×1 and 'valid' padding. The ReLU activation function is used, followed by another batch normalization layer to maintain the benefits of normalization at each layer of the network. No dropout is applied at this stage.

Next, the model includes a third Conv2D layer with 160 filters of size 2×2, a stride of 1×1, and 'valid' padding, also using the ReLU activation function. Batch normalization is again applied, followed by a dropout layer with a rate of 0.5 to further mitigate overfitting risks.

Subsequently, a GlobalAveragePooling2D layer is utilized to reduce the spatial dimensions of the tensor, transforming it into a single vector per feature map. This pooled output is then fed into a fully connected (Dense) layer comprising 500 units, with ReLU activation and L2 regularization (with a regularization factor of 0.01) to prevent overfitting and to encourage the model to learn sparse features.

Another dropout layer with a rate of 0.5 follows the dense layer, ensuring robust regularization before the final classification. The final layer is a Dense layer with 50 units and softmax activation, producing the probability distribution over the 50 output classes.

VI. Results

The results of our experiments are interesting. In particular, we noticed the slight but important difference between the three models. In general, the worse is the RawNet, which uses the raw waveform, while the best is the MFCC model which uses the MFCCs features.

TABLE 1: Test Accuracies

Model	Test Accuracy
MFCC	72% ± 3
MelNet	69% ± 3
RawNet	62% ± 3

We did precision plots for each model in order to visualize the precision of the model recognition for each class. Precision measures the ratio of true positive predictions to the total predicted positives for each class, indicating how many of the predicted instances were correct.

$$\text{Precision} = \frac{TP}{TP + FP}$$

We can see an improvement in this metrics as long as we go from the RawNet to the MFCC

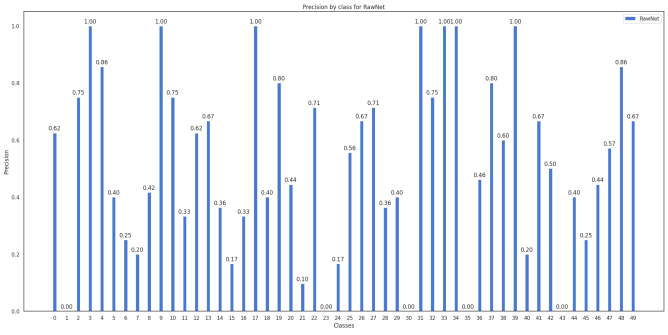

Fig. 3: RawNet Precision plot

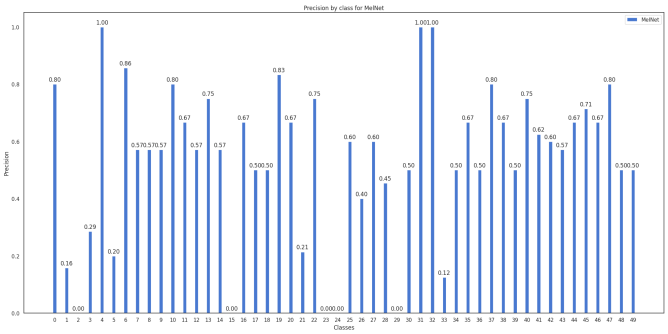

Fig. 4: MelNet Precision plot

The RawNet model appears to struggle with learning specific environmental sounds, with inconsistent performance and seemingly specializing in only a limited range of sounds. This limitation could come from its reliance on raw waveform inputs, which can introduce noise and variability that limits effective feature extraction.

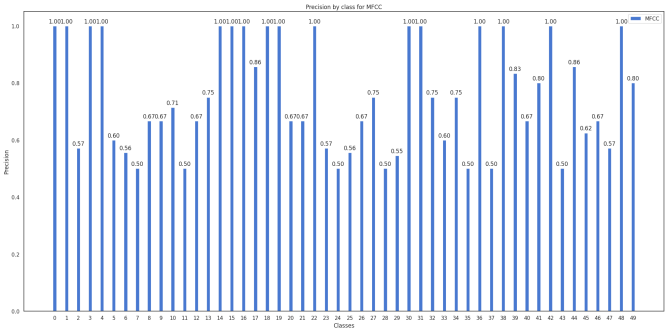

Fig. 5: MFCC Precision plot

In contrast, the MelNet demonstrates a broader capability, effectively recognizing a wider variety of sounds, even if with varying levels of precision. This improvement could be attributed to its use of log-mel features, which enhance the model's ability to capture relevant auditory patterns.

Overall, the MFCC model significantly enhances performance compared to the MelNet, establishing itself as the best model in this comparison. The success of the MFCC approach likely results from its ability to provide a more robust representation of sound characteristics, allowing for better differentiation between various environmental sounds.

VII. Concluding Remarks

In conclusion, the study on the effectiveness of different input features in CNN architectures for Environmental Sound Recognition has provided valuable insights into optimizing sound event classification tasks. The comparison of log-mel features, raw waveform input, and Mel-frequency cepstral coefficients (MFCCs) has demonstrated the significance of selecting appropriate input features for enhancing model performance.

The results indicate that the MFCC model, utilizing MFCC features, outperformed the MelNet model with log-mel features and the RawNet model with raw waveform input. The MFCC model achieved a test accuracy of 70% , showcasing its effectiveness in environmental sound recognition tasks. Precision plots further illustrated the improvement in model recognition precision from RawNet to MFCC.

In terms of future work, additional research could focus on exploring hybrid models that combine the strengths of different input features to further enhance classification accuracy. Furthermore, investigating the impact of data augmentation techniques and transfer learning on model performance could provide valuable insights for improving sound event recognition systems.

Through this project, we have been learned about the importance of feature selection, model architecture design, and performance evaluation in the context of environmental sound classification.

REFERENCES

[1] S. Li, Y. Hu, and J. Hu, "Ds-cnn: A stacked convolutional neural network for audio event recognition,"

[2] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *Read. Speech Recognit.*, 1980.

[3] O. Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Trans. Audio Speech Lang. Process.*, 2014.