"Classification problem on ESC-50 dataset" the role of attention mechanisms in Environmental Sound Classification

Anthony Palmieri†,

***Abstract*—Environmental sound classification (ESC) plays a crucial role in various applications such as soundscape analysis and acoustic monitoring. This paper explores neural network architectures for ESC, focusing on attention mechanisms. By integrating attention mechanisms and skip connections into Convolutional Recurrent Neural Networks (CRNNs), the models are able to effectively capture relevant audio features and escape suboptimal solutions. Extensive experimentation is conducted to evaluate the impact of these architectural components on model performance. The findings reveal that the integrated attention mechanisms and skip connections not only significantly enhance accuracy by enabling the models to focus on important audio features, but also offer a substantial speed-up in the fitting process. Additionally, we employ an ensemble method combining models based on raw audio waveform with models based on mel time-frequency features. The integration of these diverse model representations results in a final accuracy of 76%, showcasing the effectiveness of ensemble strategies. The results of this study provide valuable insights into the effectiveness of attention mechanisms, skip connections, and ensemble methods for ESC. The significant improvements in accuracy and convergence speed highlight the potential of these architectural components in enhancing ESC models, hopefully providing inspiration for the development of more accurate and efficient architectures.**

***Index Terms*—Environmental Sound Classification, Convolutional Neural Networks, Recurrent Neural Networks, Attention, Skip Connection, Ensemble.**

I. Introduction

The field of sound or audio recognition systems has gained significant attention in recent years, finding application in diverse domains such as intelligent audio surveillance, musical instrument classification, robot navigation, environment monitoring and so on. Sound classification involves three main domains: Music Information Retrieval (MIR), Automatic Speech Recognition (ASR), and Environment Sound Classification (ESC) or Sound Event Recognition (SER). ESC/SER specifically focuses on categorizing environmental sounds, which poses greater challenges compared to MIR and ASR due to the unstructured nature and lower Signal to Noise Ratio (SNR) of environmental audio signals. Traditional sound classification systems primarily relied on handcrafted features like Mel-frequency Cepstral Coefficients (MFCC), Gammatone features, and wavelet-based features. These features were combined with machine learning algorithms such as Support Vector Machines (SVM), Gaussian Mixture Models (GMM), and K-Nearest Neighbors (KNN) for classification purposes. However, recent advancements in deep learning, particularly Convolutional Neural Networks (CNNs), have demonstrated much more promising results in sound classification tasks. Indeed, CNNs have been successfully employed in tasks like speech recognition, music analysis, and environmental sound classification. However, while existing research in the field of ESC has examined a wide range of architectures and feature extraction techniques, the majority of these architectures consist of simple convolutional layers stacked one after the other, the true potential of recurrent neural networks (RNNs) and attention mechanisms remains largely unexplored. Motivated by this observation, the architectures we implemented aim to explore the utilization of RNNs, attention mechanisms, skip connections, and ensemble methods in greater depth. In this study, we propose novel CRNN architectures that leverage both mel-based features and raw audio data. Indeed, previous research has shown that while mel-based models generally outperform raw waveform-based models, the inclusion of the latter remains valuable as the two models tend to excel in different classes. Consequently, utilizing both raw waveform and mel-based features should enhance the discriminative power of the system. In this sense, the key contribution of this paper lies in the development of novel CRNN architectures that effectively integrate mel-based features and raw waveform by incorporating attention mechanisms. Specifically, the architectures we implemented are the following:

- A CNN architecture designed to handle raw waveform data. Given the dimensionality of the waveform, this architecture is designed to be relatively simple, incorporating many pooling layers for dimensionality reduction.
- A CRNN model designed for processing mel-based features. This model consists of an initial inception layer followed by a bidirectional GRU layer. We enhance the bidirectional layer with a trainable alignment mechanism.
- A CNN model that incorporates both attention mechanisms and skip connections. The skip connection is implemented via a convex combination guided by trainable weights.
- A final ensemble network that combines all the aforementioned models, resulting in a final accuracy of 76%.

Our final model achieves an accuracy of 76% on the test set, nearing human-level capabilities but still lagging behind state-of-the-art results. Recent studies have demonstrated that models pretrained on image datasets can surpass 97% accu-

†Department of Mathematics, University of Padova, email: {anthony.palmieri}@studenti.unipd.it

racy on the ESC-50 dataset, as evidenced in references [1] and [2]. Although we acknowledge the immense potential of such a strategy, our work does not delve into pretrained architectures due to computational and time constraints. Nonetheless, our contributions collectively demonstrate the value of incorporating attention mechanisms and leveraging ensemble methods in sound classification. The integration of attention mechanisms enhances the models' ability to focus on relevant audio features, while ensemble methods leverage the strengths of multiple models for improved accuracy. During our research, we observed a scarcity of papers that extensively utilized attention mechanisms, skip connections, and ensemble methods in sound classification tasks. Through our work, we aspire to address the existing gap in the literature and offer valuable insights into the advantages of incorporating such architectural components in the context of environmental sound classification.

This report is structured as follows. In Section II we describe the state of the art, the system and data models are respectively presented in Sections III and IV. The proposed signal processing technique is detailed in Section V and its performance evaluation is carried out in Section VI. Concluding remarks are provided in Section VII.

II. Related Work

The ESC-50 dataset was introduced for the first time in [3] as a response to the lack of suitable and publicly available datasets for environmental sound classification. In their study, the authors evaluated human accuracy in classifying environmental sounds, which was found to be approximately 81.3%. Basic machine learning classifiers such as KNN, SVM, and Decision Tree were also explored, utilizing mel-frequency cepstral coefficients and zero-crossing rate as features, averaged across each frame of the audio clip. The best-performing model was a forest ensemble with an accuracy of 44.3%. However, the paper acknowledged the limitations of these basic classifiers and emphasized the need for more powerful feature extraction techniques capable of handling multidimensional input and capturing the locality between features, which is precisely where CNNs excel.

In [4], the potential of convolutional neural networks (CNNs) in ESC-50 was evaluated for the first time. A deep model consisting of convolutional and fully connected layers was trained on segmented spectrograms, achieving an accuracy of approximately 64.5%. This performance surpassed all the basic ML classifiers tested in [3].

The work presented in [5] served as a significant inspiration for our research. The paper proposed stacked CNN models to handle both log-mel features and raw audio waveforms, which were then fused together using the Dempster-Shafer Evidence Theory to create an ensemble model. This ensemble model achieved an accuracy of about 83.1%. In our work, we propose a CNN ensemble that seems to achieve even better performance than the Dempster-Shafer theory.

In [6], a deep CNN architecture combined with Mixup augmentation on raw audio was proposed, achieving an accuracy of 83.7%. However, although Mixup improved accuracy on many classes, it also resulted in a slight performance deterioration on other classes due to audio contamination. We tested Mixup as well but later discarded it as it didn't provide substantial improvements in accuracy. We will discuss this further in subsequent sections.

The study presented in [7] introduced a frame-level attention model for ESC, focusing on semantically relevant and salient frames. They employed a convolutional recurrent neural network to learn spectro-temporal features and utilized a frame-level attention mechanism to learn discriminative feature representations. The proposed method achieved a classification accuracy of 86.1%, demonstrating the power of attention mechanisms and RNNs.

Another source of inspiration for our models was [8], which proposed the use of multiple feature channels, including MFCC, GFCC, CQT, and Chromagram, in combination with an attention mechanism. Mix-up data augmentation was employed, and the proposed CNN achieved an accuracy of 87.45%.

In [1], the authors presented a model for environmental sound classification based on log-power Short-Time Fourier Transform (STFT) spectrograms, incorporating approaches from the image domain. They explored cross-domain pre-training by initializing the model with weights learned from the ImageNet dataset before fine-tuning it on environmental sound data. Leveraging transfer learning from ImageNet, the model outperformed previous approaches, achieving an accuracy of 91.50%.

Lastly, [2] also explored the use of transfer learning with pretrained architectures on image datasets. Mel spectrogram features were used, and in particular, the pre-trained *DENSENET161* achieved an accuracy of 97.57%, representing the current state-of-the-art result as far as we know.

As we can see, pre-training and transfer learning have demonstrated significant performance advantages over other architectures, achieving state-of-the-art results. Although we acknowledge the potential of these strategies, our research will not focus on pretrained architectures due to the limitations in computational resources and time constraints.

III. Processing Pipeline

The research utilized data from the publicly available ESC50 dataset, which comprises 2000 short environmental recordings. These recordings are divided into 50 classes, evenly distributed across five major groups. To ensure consistent cross-validation, the dataset was pre-arranged into five folds. In order to accommodate the requirements of various deep learning models, the data underwent different pre-processing techniques, which will be explained in detail in the subsequent section. To address the limited data availability and mitigate overfitting, data augmentation techniques were employed. For both the models based on raw audio and mel features, the audio clips were segmented into partially overlapping clips, effectively expanding the size of the dataset. For the mel-based models, additional data augmentation was

performed by introducing random delays and pitch shifts to each segment. As for the raw model, mixup augmentation was initially explored. However, it was observed that mixup did not contribute to improved accuracy and only increased the computational burden. Consequently, mixup augmentation was excluded from the final model. Since each clip is segmented into multiple parts, predictions were made using a probability voting scheme. The highest-ranked (most probable) class across the segments generated by each clip was assigned as the final prediction. This approach harnessed the diversity of predictions from the segmented clips, resulting in a more robust and accurate classification outcome. Lastly, all predictions from the three different models were fed into an ensemble network. This ensemble model combined the outputs of the individual models, leveraging their collective strengths to enhance the overall performance. The general workflow is summarized by the following picture.

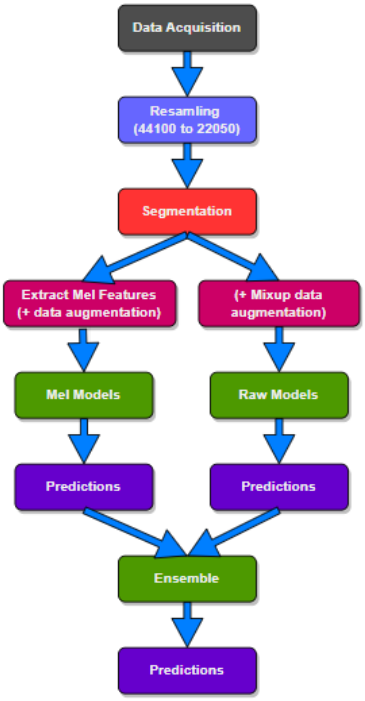

Fig. 1: project workflow.

IV. Signals and Features

The dataset used in this project consists of recordings of audio events, with each clip being 5 seconds long. The samples are presented in a unified format of 44.1 kHz, single channel, Ogg Vorbis compression at 192 kbit/s. To facilitate cross-validation, the dataset is organized into 5 folds, ensuring that clips from the same source file are always contained in a single fold. In our work, for computational reasons, the signals are resampled at 22050 Hz, resulting in raw audio waveforms of length $5 \cdot 22050$. The silence frames are retained to preserve all the information in the audio clip. Each raw waveform is then segmented into four segments of 2 seconds each, with a 50% overlap between subsequent segments. At this stage, the mixup augmentation technique was applied to the raw model. Pairs of randomly chosen segments were mixed together using a mixing factor, β, sampled from a Beta distribution $\beta(0.2, 0.2)$, as discussed in [6]. However, it was observed that mixup did not significantly improve accuracy and thus was excluded from the final model. For the mel-based model, after audio segmentation, the mel spectrogram is computed using the Librosa library in Python. The parameters used are n_fft = 1024, hop_length = 512, and n_mels = 60. The resulting mel spectrogram is then log-scaled, and both delta and delta-delta features are computed. Additionally, the first 20 Mel-Frequency Cepstral Coefficients (MFCC) and their corresponding delta and delta-delta features are computed as well. The same parameters (n_fft = 1024, hop_length = 512) are used for MFCC computation. At this point, for each segment there are two tensors: one with dimensions $(60, 87, 3)$, containing for each of the 87 frames the log-scaled mel spectrogram and its derivatives, and another with dimensions $(60, 87, 1)$, containing for each frame the $20 + 20 + 20$ MFCC and related delta and delta-delta features. These tensors were stacked together to obtain, for each segment, a tensor of dimension $(60, 87, 4)$. Data augmentation was further applied to the training set. For each segment, two additional segments are generated by introducing a random delay of up to 0.6 seconds (with the new audio truncated at 2 seconds) and a pitch shift sampled uniformly from the interval $[-3, +3]$. This data augmentation is applied before the extraction of mel features and only on the training set. Regarding the splitting of the dataset, instead of using a 5-fold cross-validation approach, an alternative hold-out approach is chosen due to computational constraints. The original dataset is divided into $Train\_Validation$ and $Test$ sets, with proportions of 80% and 20%, respectively. The $Train\_Validation$ set is then further divided into $Train$ and $Validation$ subsets, each accounting for 80% and 20% of the original size, respectively. Finally, given that each audio clip is split into several segments, predictions are made using a probability voting scheme. The highest-ranked (most probable) class across the segments generated by each clip is assigned as the final prediction, leveraging the diversity of predictions from the segmented clips to enhance robustness and classification accuracy.

V. Learning Framework

Before exploring the various implemented NN architectures in detail, let us briefly discuss the overall learning strategy employed. In the learning framework for all four models (raw, mel_1, mel_2, ensemble), we employ the Adam optimizer to minimize the categorical cross-entropy loss function. However, there are slight variations in the training procedures between the raw model, the two mel-features based models, and the ensemble network. For the raw model and the mel-features based models, the learning procedure is as follows:

- We train the models on the augmented training set and utilize the validation set to select optimal hyperparame-

ters. The models are trained for 70 epochs with a batch size of 20.

- During training, we employ two Keras callbacks. The "checkpoint" callback saves only the best model based on validation accuracy, while the "ReduceLROnPlateau" callback reduces the learning rate by a factor of 0.2 if no improvement in validation accuracy is observed for more than 3 epochs.
- After the initial training phase, we merge the training and validation sets and retrain the models on the combined $Train\_Validation$ set for another 50 epochs, using a batch size of 20.
- Similar to the previous step, we utilize the "checkpoint" callback to save the best model based on accuracy. Additionally, we apply the "ReduceLROnPlateau" callback to reduce the learning rate by a factor of 0.2 if no improvement in accuracy is achieved for more than 5 epochs.
- The obtained models are then employed to make predictions on the $Test$ set based on a probability voting scheme applied on the segments produced by each audio clip.

Recall that the input tensor for the raw model has shape $(1, 2 \cdot 22050)$, corresponding to a 2 seconds long segment resampled at 22.05 kHz. The input tensor for the mel models has shape $(60, 87, 4)$ corresponding to the $60 \cdot 4$ mel-based features extracted from the 87 frames of the segment. On the other hand, the training procedure for the ensemble network differs slightly:

- Since the ensemble model takes predictions made by models trained on the full $Train \bigcup Validation$ set, it does not make sense to use a separate validation set for hyperparameter selection. Therefore, we train the ensemble network on the full $Train\_Validation$ set.
- To limit overfitting, we train the ensemble for a limited number of epochs, specifically 5 epochs, with a batch size of 20.
- During training, we utilize the "checkpoint" callback to save the best model based on accuracy.
- Once again, $Test$ set prediction are obtained based on a probability voting scheme applied on the segmentes produced by each audio clip.

Recall that the input tensor of the ensemble has shape $(50, 3)$, consisting of the 50 predicted probabilities of the three previous models. With the learning strategy overviewed, we can now delve into the analysis of the NN architectures.

A. Raw Model

The main building blocks of the raw model are the *cba* layers presented below.

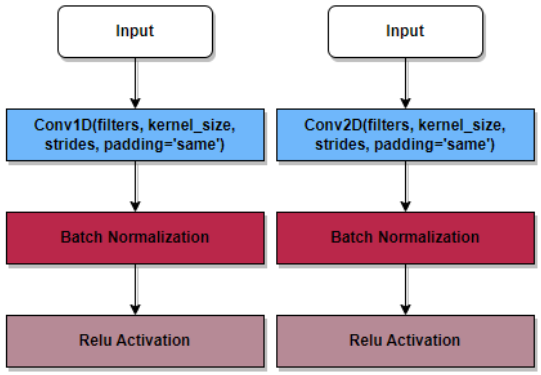

Fig. 2: CBA blocks. The convolution layer extracts meaningful features; the Batch normalization account for the internal covariance shifting, accelerating training and improving performances; Relu activation introduces non-linearity.

Using *cba1D* and *cba2D* we can build the final architecture:

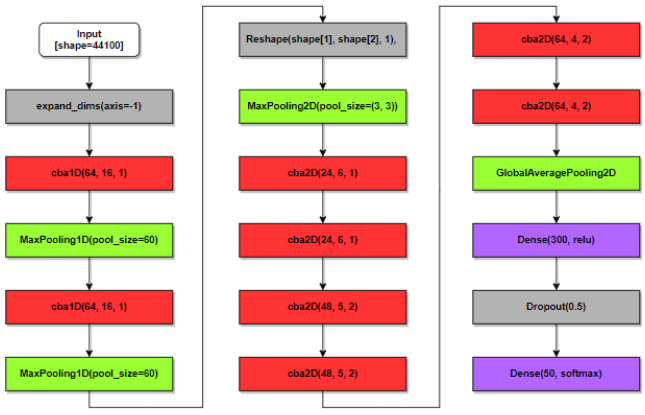

Fig. 3: raw model architecture.

This architecture consists of a combination of 1D and 2D convolutional layers, pooling layers for spatial dimensionality reduction, and fully connected layers for classification. The use of both 1D and 2D convolutional blocks allows the model to capture local patterns in the input data, enhancing its ability to extract meaningful features. The final output layer with softmax activation produces probabilities for each of the 50 classes.

B. Mel Models and Inception Layers

The mel features based models are built upon the concept of inception blocks, which serve as the fundamental building blocks for high-level feature extraction. The utilization of inception blocks enables the models to capture and extract features from the input data at multiple scales by employing convolutional layers with increasing kernel sizes. The inception blocks can capture features at different spatial resolutions, allowing for a more comprehensive representation of the input

data. The inception block employed in our architecture is designed as follows.

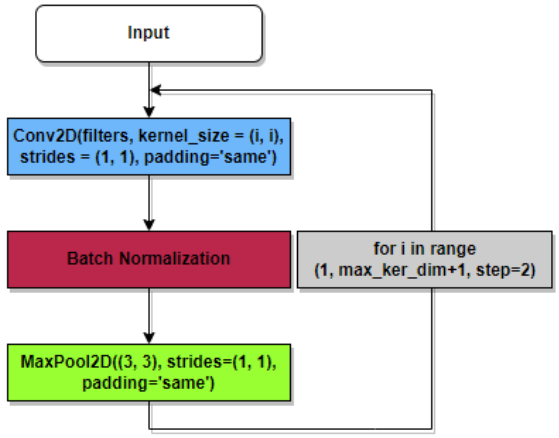

Fig. 4: inception block used for feature extraction. By leveraging the use of different receptive fields, inception blocks allow more powerful feature representation than basic convolutional layers.

By sequentially deploying layers with increasing receptive fields, our inception blocks allow the models to capture features at different scales and resolutions. The Conv2D layers extract local features, the BatchNormalization layers ensure stable and normalized activations, and the MaxPool2D layers aggregate local information. To construct the inception layer, we utilize inception blocks in the following manner:

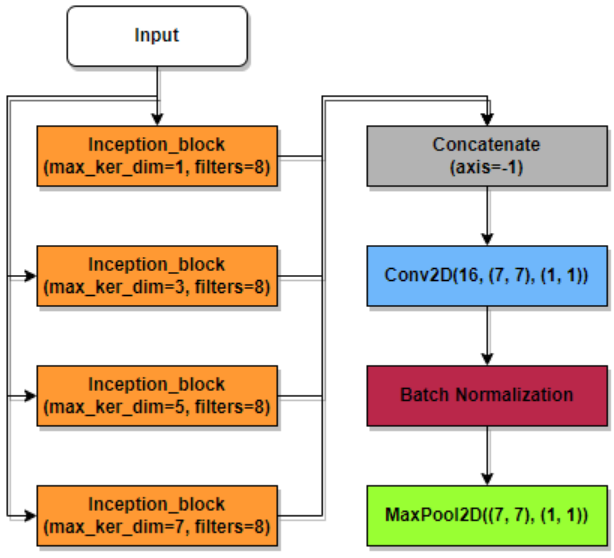

Fig. 5: inception layer, obtained by concatenating several inception blocks working in parallel.

This construction of the inception layer enables the network to capture multi-scale features and enhance the model's representation capabilities.

C. Mel_Model_1

The model architecture consists of an inception layer and a subsequent recurrent neural network (RNN) with a weighted alignment mechanism. Specifically:

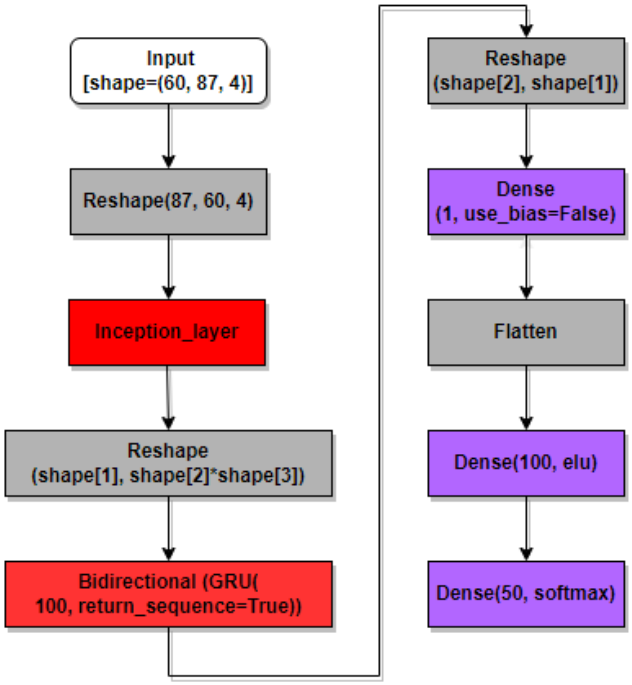

Fig. 6: first mel-features based model. It uses an inception layer for feature extraction and a subsequent RNN to capture temporal dependencies across frames. The weighted alignment mechanism helps the network focus on the most relevant frames.

This architecture combines the feature extraction capabilities of the inception layer with the ability of the RNN to capture temporal dependencies, while the weighted alignment mechanism helps the network focus on the most relevant frames for classification.

D. Mel_Model_2

The second mel model incorporates a different attention mechanism combined with skip connections to improve performance. The model is composed of three main building blocks: *Main* block, *Attention* block, and *sigma_CNN* block. Let's explore each block in detail.

1) Main block: The *Main* block is responsible for extracting relevant features from the input data. It utilizes inception blocks with increasing kernel sizes (1, 3, and 5) to capture spatial information at different scales. The outputs of these inception blocks are concatenated together. A 3×3 convolutional layer with 4 filters is then applied, followed by batch normalization, ReLU activation, and max pooling. This block efficiently extracts informative features from the input.

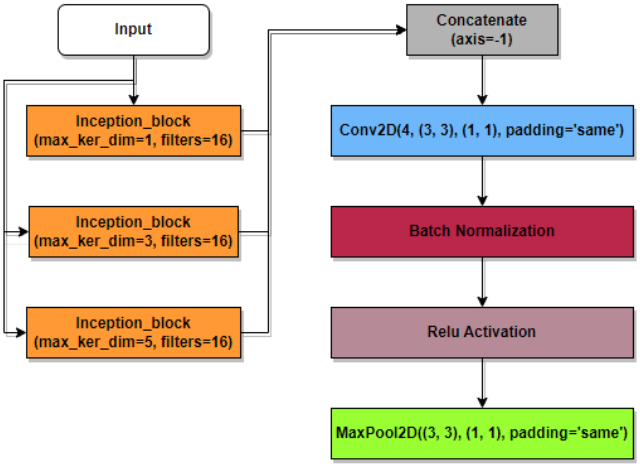

Fig. 7: *Main* block. By deploying inception blocks with increasing kernel sizes, this block effectively extracts informative features from the input.

2) Attention block: In the *Attention* block, various types of attention are computed to weigh the output of the *Main* block. The block consists of three convolutional layers with different kernel sizes: (1, 5) for spatial attention on the frequency domain, (5, 1) for temporal attention on the time domain, and (5, 5) for time-space attention on the time-frequency domain. These attention scores are concatenated to obtain the final attention representation. A 1×1 convolutional layer with 4 filters is applied, followed by batch normalization and sigmoid activation.

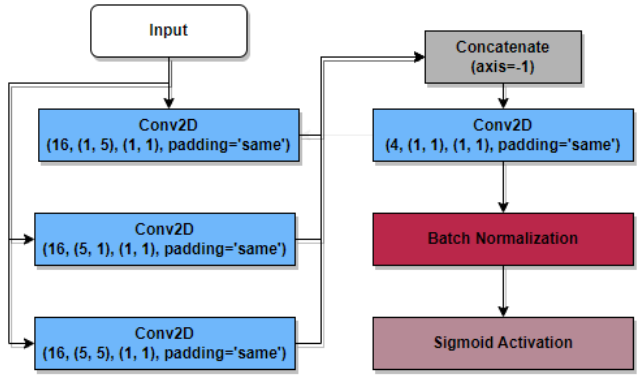

Fig. 8: *Attention* block. By using convolutional kernels of sizes (1, 5), (5, 1), (5, 5) this block extract spatial, temporal and space-time information from the original input. We then use such information to decide which parts of the input should exert more influence for the classification.

The combination (point-wise multiplication) of *Main* block and *Attention* block results in a transformed input where the relevant features are amplified based on the computed attention scores.

3) sigma_CNN block: To control overfitting and avoid gradient problems, we include a skip connection guided by the *sigma_CNN* block. Such block takes as input both the original *input* (i.e. X_{input}) and *Main(input)*$\odot$*Attention(input)* (i.e. X_{new}), reduces the dimensions of the tensors using average pooling and convolutional layers and then concatenates them. The final tensor is passed through a dense layer with a single unit and sigmoid activation function.

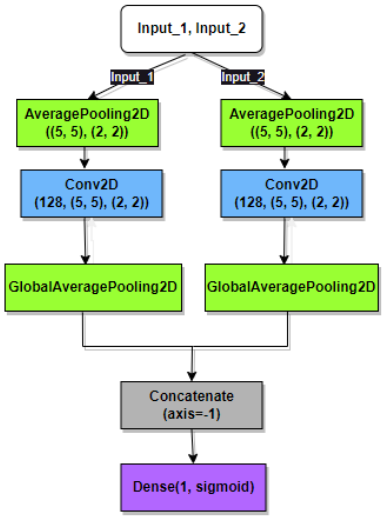

Fig. 9: *sigma_CNN* block. This layer implements another attention mechanism to efficiently guide the model through the skip connection. The final sigmoid activation returns a scalar value σ in the range $(0, 1)$, which we use as convex multiplier for the skip connection.

This layer computes a convex multiplier σ, which allows the network to dynamically adjust the contribution of each component in the skip connection $\sigma X_{new} + (1 - \sigma) X_{input}$, leading to accelerated training and improved accuracy.

We can finally build the second mel model:

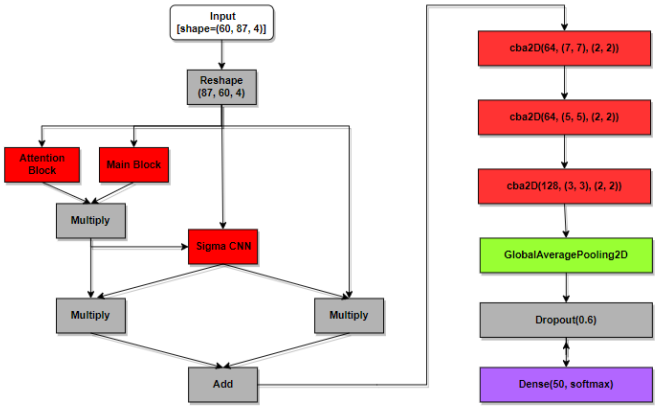

Fig. 10: second mel-features based model. It uses attention mechanisms and skip connections to effectively guide the model through learning. After the skip connections, various CBA blocks are applied to reduce dimensionality while extracting useful features.

E. Ensemble Model

The ensemble model is designed to leverage the predictions from multiple models, combining their strengths and compensating for their weaknesses, ultimately leading to more robust classification performance. Given the limited amount of data available, the model architecture is intentionally kept simple to avoid overfitting and control the number of parameters. The ensemble model consists of a single Conv1D layer with 128 filters, a kernel size of 50 and no bias. We follow with a global average pooling layer to reduce the dimension of the tensor. The output from the pooling layer is then fed into a dense layer with 50 neurons and softmax activation.

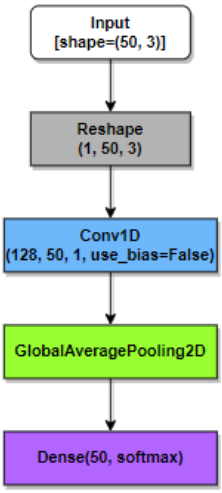

Fig. 11: ensemble network. It combines the three previous models by fusing their outputs. To enhance the flexibility of the ensemble, a 1D convolutional layer with a stride of 50 is introduced. This layer introduces a trainable weight for each probability score from the three models. This flexible weighting mechanism enables the ensemble to improve the accuracy of the individual models by dynamically adjusting the contribution of each model's output.

VI. Results

In this section, we present the results obtained from evaluating our models. Accuracy was chosen as the evaluation metric since all classes in the dataset are perfectly balanced, ensuring an equal representation of each class. Let's begin with the raw model.

A. Raw Model

The raw model achieved an accuracy of 60.75% on the test set, with a mean time per epoch of 24 seconds. It consists of 324,998 trainable parameters. During training, we observed that the model's accuracy on the validation set reached a plateau at around 56% within the first 40 epochs, indicating limited improvement. The raw model struggled across multiple classes and emerged as the weakest among the implemented models. Examining its architecture, we notice the presence of pooling layers with large pool sizes. This design choice was driven by the length of the input waveform, allowing dimensionality reduction before applying more complex layers. However, this approach resulted in significant information loss, adversely affecting the model's accuracy.

B. Mel_Model_1

Next, we evaluate the first mel model, which achieved an accuracy of 70.5% on the test set, with a mean time per epoch of 24 seconds. This model contains 580,793 trainable parameters. Similar to the raw model, the accuracy of the first mel model also plateaued within the first 40 epochs, showing minimal improvement. Initially, we experimented with a basic dot-product alignment mechanism, which yielded satisfactory accuracy. Out of curiosity, we tested a weighted attention mechanism, expecting only marginal enhancements. Surprisingly, the incorporation of weighted attention led to significant improvements in both convergence speed and accuracy, with just 200 additional parameters. Additionally, the inclusion of the inception layer proved essential, as stacking simple convolutional layers together resulted in poor performances (around 50% accuracy).

C. Mel_Model_2

Moving on to the second mel model, it achieved an accuracy of 71.5% on the test set, with a mean time per epoch of 18 seconds. This model contains 237,723 trainable parameters. The initial version, consisting of only the *Main* block and *Attention* block, encountered a numerical instability issue during training, resulting in a loss function that became NaN around an accuracy of 5%. To address this problem, we introduced skip connections, which successfully helped the model escape the suboptimal situation. However, the skip connections significantly slowed down the fitting process. To mitigate this challenge, we introduced a trainable convex multiplier σ to dynamically adjust the contribution of each component in the skip connection. This approach accelerated training and improved accuracy. It is worth noting that skip connections also posed fitting challenges in several other architectures we tested, not limited to this model.

D. Ensemble

Considering that each model struggled with different classes, we developed an ensemble method to harness their diverse strengths. The ensemble model, with a simple architecture and 25,650 parameters, aggregated predictions from the individual models. Remarkably, the ensemble model achieved an accuracy of 76% on the test set, surpassing the accuracy of the best individual model by almost 5%.

E. Final Observations

It is worth mentioning that we intentionally designed all architectures to have the minimum number of parameters required to achieve satisfactory accuracies, considering the limited data available. We carefully balanced the trade-off between model complexity and performance, and we discovered that adding more filters to the convolutional layers did not yield any improvements and only resulted in slower convergence. We also experimented with dilated convolutions

and depthwise convolutions, but these approaches reduced the number of parameters excessively and led to underfitting on the training set.
Furthermore, audio segmentation played a crucial role in enhancing the models' performance. Without segmentation, there was a significant drop in accuracy across all models. We also experimented with mixup as an additional data augmentation strategy for the raw model. However, mixup did not yield substantial improvements, leading us to discard it to alleviate memory load. Memory management proved to be a challenging task due to the limited 12GB RAM in the free version of Colab, necessitating the extensive utilization of Google Drive to store datasets and promptly delete them from the Colab working memory when no longer needed.

VII. Concluding Remarks

In this study, we conducted a comprehensive exploration of neural network architectures for environmental sound classification, specifically focusing on the integration of attention mechanisms, skip connections, and ensemble methods. We examined the influence of these components on overall accuracy and convergence speed. Our findings revealed that a trainable alignment mechanism significantly accelerated model fitting and slightly improved the accuracy of our CRNN architecture. Conversely, we observed that skip connections, while allowing the model to escape suboptimal solutions, dramatically slowed down convergence. In this sense, introducing trainable parameters to guide the skip connections proved crucial, substantially improving both performance and speed. We also assessed the impact and feature extraction capabilities of inception layers, which proved essential in achieving good performance for the mel-features based model we tested.
To address the limited availability of data, audio augmentation strategies were employed. Audio segmentation proved to be highly effective, significantly improving the performance of all tested models. We also explored other augmentation techniques like delay addition, pitch shift, and mixup on individual segments. We found that combining segmentation and mixup led to a decline in performance, likely due to the information loss caused by segmentation and contamination induced by mixup.
Overall, our investigation highlights the potential of integrating attention mechanisms and skip connections in sound classification, enhancing accuracy by focusing on relevant audio features. Moreover, leveraging a neural network ensemble method showed superior performance compared to simpler ensemble methods based on probability averaging.
Our research addresses gaps in the literature, providing valuable insights into the effectiveness of attention mechanisms and skip connections in the field of ESC.
Future avenues for refinement could include exploring transfer learning from pre-trained architectures on image datasets, investigating alternative feature extraction techniques like autoencoders, incorporating additional data augmentation strategies such as time-stretch and noise injection for improved robustness, and maybe also enhancing the alignment mechanism through the introduction of more weights.

A. What I learned

Working on this project has been a great learning experience. It was my first time diving into audio signal processing, and I found it fascinating to explore the unique characteristics and complexities of audio within the machine learning realm. I am particularly satisfied with the progress I made in utilizing Keras and TensorFlow. It allowed me to develop proficiency in implementing custom callbacks and constructing more intricate architectures beyond simple layer stacking.
Furthermore, I had the opportunity to witness the tangible impact of skip connections and attention mechanisms on the learning process. These components significantly influenced both the accuracy of the models and the speed at which they converged. It was surprising to see how adding only 200 trainable alignment weights completely changed the fitting process, making it extremely faster. This showed me that even a few carefully placed parameters can have a significant impact on learning.
The major challenge I encountered was effectively managing memory, which was a new aspect for me. Unlike previous projects with smaller datasets, the limited memory capacity had noticeable consequences on the architectural choices I had to make. These memory constraints heightened my awareness, particularly during the initial stages of coding, as exceeding memory limits often necessitated runtime restarts.
Overall, working on this project has been a rewarding experience, providing valuable insights into the use of some advanced deep learning techniques.

References

[1] A. Guzhov, F. Raue, J. Hees, and A. Dengel, "Esresnet: Environmental sound classification based on visual domain models," Apr. 2020.
[2] Z. Mushtaq, S.-F. Su, and Q.-V. Tran, "Spectral images based environmental sound classification using cnn with meaningful data augmentation," *Applied Acoustics*, vol. 172, Aug. 2020.
[3] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *ACM international conference on Multimedia*, Apr. 2018.
[4] 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, Sept. 2015.
[5] 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*, July 2018.
[6] Z. Zhang, S. Xu, S. Cao, and S. Zhang, "Deep convolutional neural network with mixup for environmental sound classification," Aug. 2018.
[7] Z. Zhang, S. Xu, T. Qiao, S. Zhang, and S. Cao, "Attention based convolutional recurrent neural network for environmental sound classification," Sept. 2020.
[8] J. Sharma, O.-C. Granmo, and M. Goodwin, "Environment sound classification using multiple feature channels and attention based deep convolutional neural network," Aug. 2020.
[9] Z. Mushtaq and S.-F. Su, "Environmental sound classification using a regularized deep convolutional neural network with data augmentation," *Applied Acoustics*, vol. 167, p. 107389, Oct. 2020.
[10] X. Zhang, Y. Zou, and W. Shi, "Dilated convolution neural network with leakyrelu for environmental sound classification," pp. 1–5, Aug. 2017.