ACOUSTIC SCENE CLASSIFICATION FROM BINAURAL SIGNALS USING CONVOLUTIONAL NEURAL NETWORKS

Technical Report

Rohith Mars, Pranay Pratik, Srikanth Nagisetty, Chong Soon Lim

Panasonic R&D Center, Singapore
{rohith.mars, pranay.pratik, srikanth.nagisetty, chongsoon.lim}@sg.panasonic.com

ABSTRACT

In this report, we present the technical details of our proposed framework and solution for the DCASE 2019 Task 1A - Acoustic Scene Classification challenge. We describe the audio pre-processing, feature extraction steps and the time-frequency (TF) representations used for acoustic scene classification using binaural audio recordings. We employ two distinct architectures of convolutional neural networks (CNNs) for processing the extracted audio features for classification and compare their relative performance in terms of both accuracy and model complexity. Using an ensemble of the predictions from multiple models based on the above CNNs, we achieved an average classification accuracy of **79.35%** on the test split of the development dataset for this task and a system score of **82.33%** in the Kaggle public leaderboard, which is an improvement of $\approx 18\%$ over the baseline system.

***Index Terms*—** DCASE 2019, acoustic scene classification, convolutional neural networks, binaural signals.

1. INTRODUCTION

Humans perceive their surroundings primarily through the visual and audio cues presented to their eyes and ears, respectively. Though visual stimuli provide a substantial amount of information regarding the scene, it is inarguable that audio cues also play a vital role in determining the type of the environment we are immersed in. For example, an immersive experience through virtual reality (VR) is deemed satisfactory only when the associated audio aligns with the visual scene. In a simpler scenario, a person standing near a beach with eyes closed can easily infer that they are near the shore from the sound of the waves hitting the rocks or from the sound of the seagulls. It is easy to conclude that acoustic characteristics of certain environments have their own unique signature, which aids humans in distinguishing an audio scene from another.

The objective of acoustic scene classification is to empower a machine to automatically recognize the audio scene from the audio signals they are provided with. The advancement of deep learning algorithms along with availability of large datasets and increase in computational power has helped to address the acoustic scene classification challenge to a great extent. The DCASE 2019 challenge [1] consists of a total five separate challenge tasks, with Acoustic Scene Classification being one among them. This task is further divided into three subtasks, wherein we participate in the DCASE 2019 Task 1A. In this subtask, the development data and evaluation data are obtained from same the recording device.

We begin this report by describing our audio pre-processing, feature extraction and data augmentation steps in Section 2. In Section 3, we provide details on the CNN architectures used. The details of the challenge database and our results are provided in Section 4. Finally, the conclusions are presented in Section 5.

2. FEATURE EXTRACTION & DATA AUGMENTATION

In this section, we describe the audio pre-processing steps as well as the binaural audio feature extraction process. The extracted features are then provided as input to the convolutional neural network for predicting the acoustic scene class. In addition, we also discuss the data augmentation step used to improve the model's generalization to unseen data.

2.1. Binaural audio feature extraction

In our proposed system, we use the originally provided audio data sampled at 48 kHz without down sampling. The time domain audio signals are then normalized by amplitude and then converted to the time-frequency (TF) representation to extract the temporal and spectral characteristics. As such, we first compute the short-time Fourier transform (STFT) of the normalized time-domain audio signal. The frame size for the STFT operation is fixed at 2048, with a frame overlap of 50% and hanning window. Due to the large dimension of the linear STFT representation, we further compute the corresponding Mel-spectrogram representation using 128 Mel-bands. The use of Mel-scale is more close to the human auditory system and provides additional advantage of having lower dimension than conventional linear STFT. As the final step, we compute the $\mathrm{Log}(\cdot)$ of the Mel-spectrogram to reduce the dynamic range of the values and also make the feature space more Gaussian in distribution as reported in [2]. On the computed Log Mel-spectrograms, we performed feature normalization to achieve zero mean with unit variance. This mean and standard deviation was computed using the training data and the same were used on the validation/test split.

Since the recorded data set in this task is binaural audio, the above Log Mel-spectrogram is computed separately for the Left (L), Right (R), Mid (M), Side (S) representation. In addition, we also use the conventional mono representation of the binuaral signal for computing the Log Mel-spectrogram. The Mid and Side representation of the L & R signals are computed as follows

$$\begin{aligned} M &= (L+R)/2 \\ S &= (L-R)/2. \end{aligned} \tag{1}$$

The use of these binaural representations for audio classification have been explored in the earlier DCASE audio scene classification challenges [3] and has been reported to achieve superior performance. However, compared to [3] we do not split the 10 second

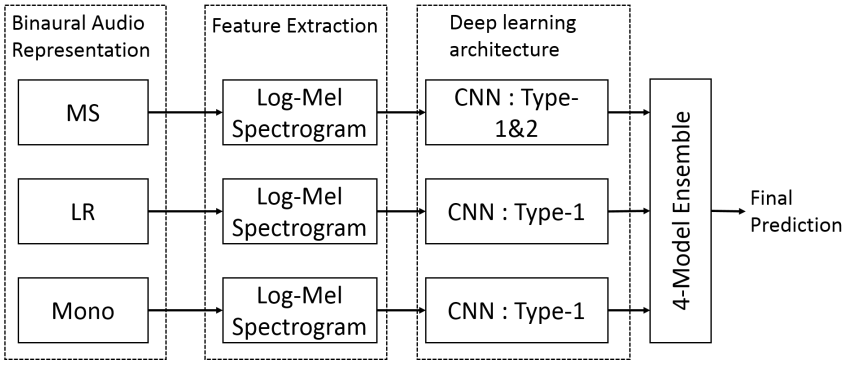

Figure 1: Our overall solution framework consisting of binaural audio representation, feature extraction steps, multiple CNNs & model ensembling for final prediction of the acoustic scene.

audio clips to smaller audio chunks. The obtained 2D spectrogram of size 128×469 per channel is provided as input to the neural network. In addition, instead of using each channel as a separate input to the CNN and concatenating the CNN outputs at a later stage, we combine the channels at the first layer of convolution itself. To further simplify the feature extraction step, we do not perform the background subtraction (BS) method as well as the harmonic percussion source separation (HPSS) based audio representations reported in [3]. The entire framework of our solution depicting the audio representations, feature extraction, multiple CNNs and ensembling step is shown in Figure 1.

2.2. Data Augmentation

It is well-known that deep learning algorithms perform well when they are trained on large amounts of data. However, depending on the task, the amount of labelled data for training maybe limited or constrained. This may result in the deep learning algorithms not to fully capture the intra-class and inter-class variations in the data. In such situations, data augmentation plays a crucial role to increase the amount of training data. For acoustic signals, the conventional augmentation include pitch shifting, time stretching, noise addition, dynamic range modulation etc. Another variation of similar augmentation is to mix the clips of same acoustic class by splitting and shuffling. Recently, the use of generative adversarial networks (GANs) for data augmentation has also been explored in [4].

In our proposed method, to ensure a better generalization capability for the neural network, we perform the augmentation method proposed in [5], termed as *mixup*. In this method, two training examples are weighted and mixed together along with their class labels to form virtual training examples as

$$\begin{aligned} \tilde{x} &= \lambda x_i + (1-\lambda) x_j \\ \tilde{y} &= \lambda y_i + (1-\lambda) y_j, \end{aligned} \tag{2}$$

where (x_i, y_i) and (x_j, y_j) are two samples drawn at random from the training set, with $\lambda \in [0, 1]$ acquired by sampling from the β distribution $\beta(\alpha, \alpha)$, with α being a hyper parameter.

3. CNN ARCHITECTURE

The audio features extracted by the pre-processing steps explained in Section 2 are provided as input to a convolutional neural network (CNN). CNNs have been vastly used for audio scene classification. They have the ability to learn both spectral and temporal patterns of the audio features. In our work, we experiment with two separate architectures of CNN. The first CNN architecture is similar to the VGG-style architecture, which uses a constant (3×3) square-shaped kernels. However, we use significantly less number of convolution and dense layers as compared to the original VGG architecture. In the second CNN architecture, we employ rectangular kernels for convolution. CNNs with rectangular kernels have been previously used for speech and music genre classification. The use of such rectangular kernels help to treat the spectral and temporal components of the audio separately. In the following subsections, we further elaborate on the above two CNN architectures.

3.1. CNN: Type-1

This CNN architecture is similar to the VGG-style architecture. It consists of 5 convolutions with increasing number of filters, i.e., $\{64, 128, 256, 512, 512\}$. The kernel size is chosen as (3×3) and is kept constant for all the convolution layers. We also apply batch normalization and ReLU non-linear activation for each convolution layers. Max pooling (2×2) operation is performed at the first three convolution layers to reduce the dimensionality. Finally we perform global max pool operation to gather all the components, which is then connected to a dense layer of 256 units with ReLU activation. The output layer consists of 10 units corresponding to the number of scene classes and undergoes softmax operation to obtain the prediction probabilities. The CNN: Type-1 has ≈ 4 million parameters.

3.2. CNN: Type-2

In this CNN architecture, we employ rectangular kernels instead of square kernels. It consists of 4 convolutions with increasing number of filters, i.e., $\{64, 128, 256, 512\}$. For the convolution layer-1, we apply a kernel of size (3×7), followed by a max pooling with size (3×1). After reducing the dimension in the frequency axis, convolutions with kernel size (3×1) is applied in the next convolution layer. We further reduce the dimension in the frequency axis by using (4×1) max pooling. In the layer-3 convolution, we use a kernel size of (11×1) and perform "valid" convolutions. This step compresses the entire frequency axis. The last convolution layer uses filter size of (1×7) to learn the temporal characteristics. Note that we have not performed pooling across time dimension. Similar to

CNN-1, we apply batch normalization and ReLU non-linear activation for each convolution layers. After the convolution layers, all the components are collected using the global max pooling operation, which is further input to a fully connected layer of 256 units with ReLU activation. The output layer consists of 10 units corresponding to the number of scene classes and undergoes softmax operation to obtain the prediction probabilities. The advantage of using this CNN architecture is in its complexity. The CNN: Type-2 uses $\approx$ 1.4 million parameters. This is 4 times lower number of parameters as compared to CNN: Type-1 and therefore, is a less-complex (light-weight) architecture.

Note that for both LR & MS representations, the input size is $128 \times 469 \times 2$, with each channel arranged back to back. For the case of mono representation, the input size is $128 \times 469 \times 1$.

4. DATABASE & RESULTS

The TAU Urban Acoustic Scenes 2019 dataset for this task is the extension of the 2018 TUT Urban Acoustic Dataset, consisting of binaural audio recordings from various acoustic scenes in different cities. The recordings were made using the Soundman OKM II Klassik/studio A3, electret binaural microphone and a Zoom F8 audio recorder using 48 kHz sampling rate and 24 bit resolution so that the recorded audio is very similar to the sound that reaches the human ears. For each acoustic scene class, such recordings were collected from different locations in the city. Each original recordings were split into segments with a length of 10 seconds as development and evaluation set. The audio scenes are namely {"Airport", "Indoor shopping mall", "Metro station", "Pedestrian street", "Public square", "Street with medium level of traffic", "Travelling by a tram", "Travelling by a bus", "Travelling by an underground metro" & "Urban park"}.

From the training split of the development set, we use a random split of 15% as the hold-out validation set. The optimization is performed using the Adam optimizer, with an initial learning rate of 0.001 and a maximum epoch of 200 with a batch size of 32 samples. We reduce the learning rate if the validation loss does not decrease after 5 epochs. We use early stopping method if the validation loss does not decrease after 10 epochs. The categorical cross-entropy is chosen as the loss function. For the data augmentation step using *mixup*, we kept $\alpha = 0.3$. The baseline system also used a CNN based approach on Log Mel spectrum of 40 bands, consisting of two CNN layers and one fully connected layer. We chose Keras as our deep learning framework for all experiments.

For the CNN: Type-1, using the mono representation, we get an average accuracy of 73.04%. In comparison, the MS and LR representation, we achieve an accuracy of 75.19% and 74.2%, respectively. Using the ensemble of MS, LR mono representations, we get an accuracy of 78.5%. For CNN: Type-2, the accuracy are 69.86%, 72.9% & 69.79% using the mono, MS and LR representation, respectively. The performance drop maybe due to the low-complex architecture used in CNN: Type-2. As such, we select the top-4 best performing models for the final ensembling. We ensemble the predictions from each of the 4 models by computing the geometric mean of the predictions. The final prediction is done by selecting the class with maximum probability on the ensembled prediction. After this ensembing, we obtain an accuracy of **79.35%** on the test split of the development set. In the Kaggle public leaderboard, the same solution achieved a system score of **82.33%**. The classification results compared with the baseline system are shown in Figure 2. The confusion matrix obtained we obtained using the ensembled model is shown in Figure 3.

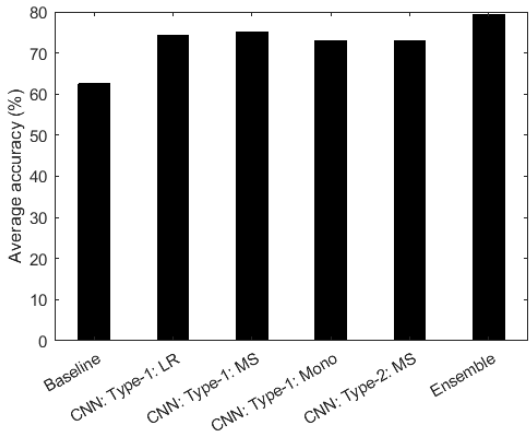

Figure 2: Average classification accuracy for the baseline system as well as each of our proposed models.

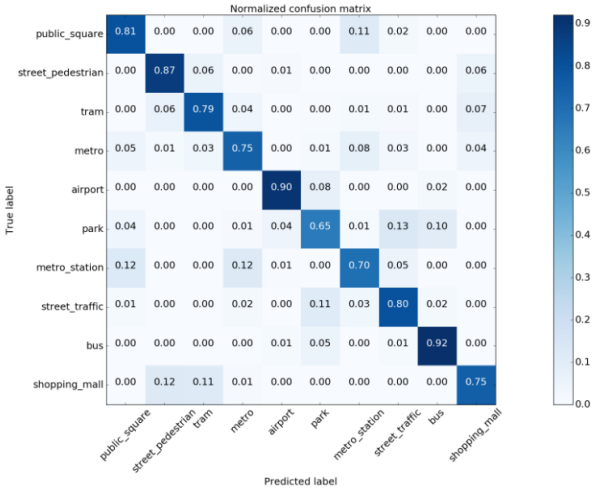

Figure 3: Confusion matrix of the proposed system after ensembling.

5. CONCLUSIONS

In this report, we provided the details of our solution to the DCASE2019 Task1A - Acoustic Scene Classification. We described the audio pre-processing, feature extraction steps and the various binaural audio feature representations used as input the neural network. The architecture of the convolutional neural networks used for the classification task are described. Our solution achieves an average accuracy of **79.35%** on the test split from the development set. The same solution achieved a system score of **82.33%** in the Kaggle public leaderboard, which is an improvement of $\approx 18\%$ over the baseline system.

6. REFERENCES

[1] http://dcase.community/challenge2019/.

[2] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "A comparison of audio signal preprocessing methods for deep neural networks on music tagging," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1870–1874.

[3] Y. Han, J. Park, and K. Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," *the Detection and Classification of Acoustic Scenes and Events (DCASE)*, pp. 1–5, 2017.

[4] S. Mun, S. Park, D. K. Han, and H. Ko, "Generative adversarial network based acoustic scene training set augmentation and selection using svm hyper-plane," *Proc. DCASE*, pp. 93–97, 2017.

[5] H. Zhang and et.al, "mixup: Beyond empirical risk minimization," *International Conference on Learning Representations*, 2018.