ADVANCED AUDIO PROCESSING - COMP.SGN.220-2020-2021-1

Acoustic Scene Classification with Multiple Devices

(DCASE 2020 - Task 1a)

PROJECT REPORT

Duy Vu - H292118 - duy.vu@tuni.fi

An Tran - H281786 - an.tran@tuni.fi

Trung Kien Duong - H272573 - kien.duong@tuni.fi

The source code and setup of the data processing and model training can be found at:
https://github.com/haantran96/advanced-audio

1. Introduction

Acoustic Scene Classification (ASC) is the task of classifying the location where the audio is recorded. Each audio corresponds to one target class (in total of 10 classes), which specifies which scenes the recordings take place. The 10 present locations are airport, shopping mall, metro station, street pedestrian, public square, street traffic, tram, bus, metro, park. There are no cases of multi-labeled data [1]. ASC was released in two subtasks A and B. This report works only in Subtask A, which consists of 64 hours of audio recorded in 10 different European cities using 9 (real and simulated) microphone devices. We employed a ResNet fusion model scheme that learns features from different frequency bins and merges the learnable features to predict the target class. Without any augmentation, but using data that are not originally included in the train segments of the cross-validation setup, in other words, 20,070 audio segments instead of 13965, the model was able to achieve the accuracy of around 77% on development test dataset.

2. Data pre-possessing

2.1. Dataset: TAU Urban Acoustic Scene 2020 Mobile

The Development set of TAU Urban Acoustic Scene 2020 Mobile consists of 23,040 ten-second audio samples collected in different settings. First, the recordings are conducted at 10 environments, including airport, shopping mall, metro station, street pedestrian, public square, street traffic, tram, bus, metro, park in different European cities. Second, the devices used for data collection includes three real (i.e binaural microphone, smartphones) and six simulated ones. The simulated recorders are synthesized by processing real devices' samples with impulse responses and dynamic range compression. A metadata file for all the samples is given, including scene_label (10 different environments where the recordings are taken), identifier, and source_label (6 devices, 3 real devices A,B,C and 6 synthetic ones S1-S6). In addition, the authors also provide the training/test partitions on the development, with 13,965 samples in the training set and 2,970 samples on the test set [1].

There is also an Evaluation split for the dataset, which consists of 11,880 audio samples recorded by GoPro Hero5 Session device. Since the Evaluation set is made private and restricted to DCASE 2020 participants, our model is evaluated based on the 2,970 audio samples from the Development dataset's test partition [1].

2.2. Feature Extraction

The input data are mono ten-seconds audio files with 44,1kHz sampling rate. The raw audios are processed in three ways. First, log-mel spectrogram features are extracted from the audio waveforms using a window size 2048, hop length 1024 (50%), Hann window function and 128 or 256 Mel bins. The log-mel energies are then normalized using min max scaling. The obtained

spectrogram has a shape of [mel_bins, 431] [2]. From the log-mel spectrogram, we compute deltas and delta-deltas features. The delta is calculated by:

$$\Delta_k = f_k - f_{k-1} \quad [4] \tag{1}$$

The delta-delta is simply the second difference, defined as:

$$\Delta\Delta_k = \Delta_k - \Delta_{k-1} \quad [4] \tag{2}$$

The log-mel energies, deltas and delta-deltas are stacked along the channel axis. The obtained feature has shape [3 x mel_bins x time_steps], where each channel represents different features. The number of frames is reduced little bit during the calculation of delta-deltas feature.

2.3. Training and validation sets

From the audio samples in the given training partition, we randomly split the data into training and validation subsets, with 85% allocated for training and 15% allocated for validation.

3. Model Setup

3.1. Model architecture

We propose a N-ResNet model, which is a fusion of multiple ResNet models applied on different Mel ranges of the input feature. Figure 1 illustrates the general architecture of N-ResNet.

Specifically, we experiment with 2-ResNet and 3-ResNet architectures, depending on how many frequency bin features were extracted on the Mel frequency scale. For 3-ResNet, we used 256 Mel bands, resulting in the shape of [3 x 256 x Time]. Then, 3 parallel ResNet18 models are applied in the 3 sub-segments: [3 x 0:64 x Time], [3 x 64:128 x Time] and [3 x 128:256 x Time] [3]. For 2-ResNet, we used 128 Mel bands, and applied 2 ResNet18 on 2 parts [3 x 0:64 x 423] and [3 x 64:128 x 423] [2].

The learned features are then concatenated and forwarded through 2 Convolution 1x1-Batch Normalization-ReLU blocks, and finally average pooling by the two last axes to obtain the classification score. The resulting vectors of the architecture should be of shape [Batch_size x N_classes], where in our cases N_classes=10 representing the total number of target classes.

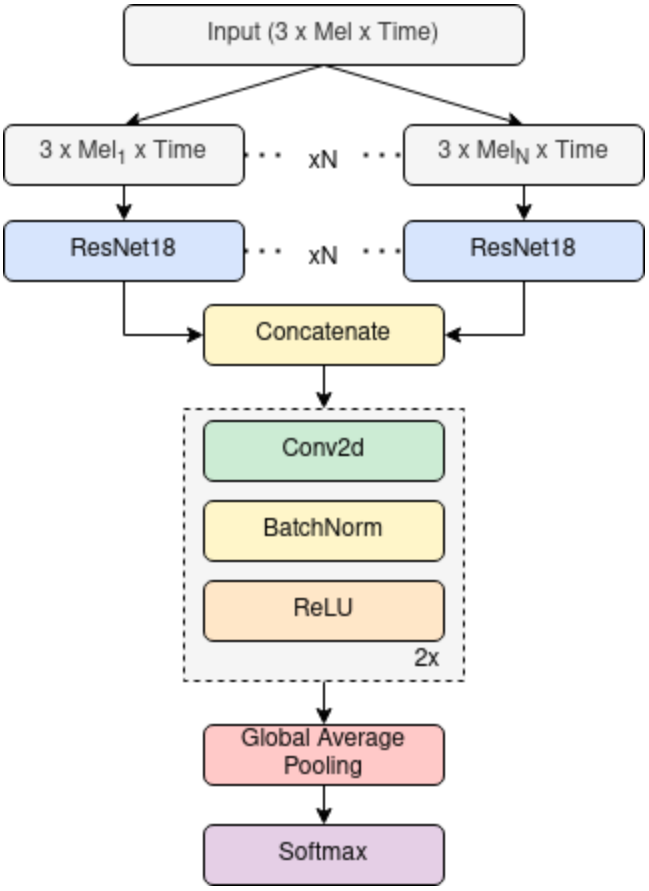

Figure 1: N-ResNet model

The ResNet model is chosen for its effectiveness in multiple tasks, including image classification, video recognition, and audio recognition. In ASC, ResNet is proved to work effectively [5, 6]. By splitting the input by Mel bins, the parallel models are expected to learn distinct features from the frequency bins and aggregate more useful information [7].

3.2. Training and hyperparameters setup

In the training phase, we split the audio samples into 85/15 splits for training and validation. The training is conducted over 200 epochs, with an early stopping policy which stops the training if results do not improve after 20-30 epochs. We employed Adam optimizer, with learning rate 2e-4, weight decay 1e-4 and ReduceLROnPlateau scheduler to adjust the learning rate.

Figure 2 and 3 show the loss and validation of 2-ResNet and 3-ResNet models over the epochs. Overall, the 3-ResNet model converges on the validation split while 2-ResNet is unstable over the epochs.

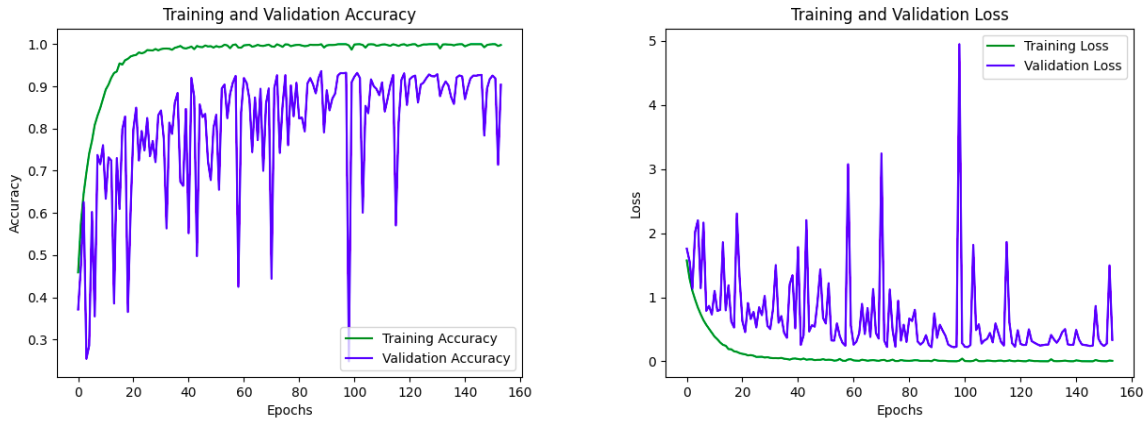

Figure 2: 2-Resnet

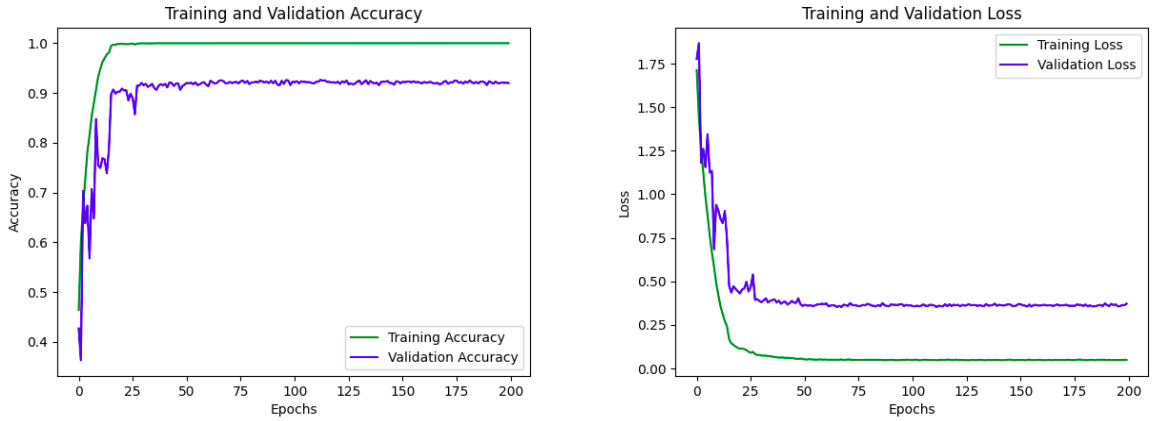

Figure 3: 3-ResNet

4. Result

Since the Evaluation set is made private, we evaluated our models on the available testing partition from the Development set. DCASE's baseline model is also evaluated on this set. The test partition consists of 2,970 audio samples, some of which are recorded by different devices from the training split, which introduces more variation between the recordings.

Scene class	**Baseline**	**2-ResNet**	**3-ResNet**
Airport	45.0%	**75.8%**	68.0%
Bus	62.9 %	**93.3%**	85.9%
Metro	53.5 %	80.1%	**84.5%**
Metro station	53.0 %	77.8%	**81.1%**
Park	71.3 %	80.1%	**84.1%**
Public square	44.9 %	51.5%	**51.9%**
Shopping mall	48.3 %	78.5%	**82.2%**
Street, pedestrian	29.8 %	54.5%	**63.0%**
Street, traffic	79.9 %	92.3%	**93.0%**
Tram	52.2 %	63.2%	**82.4%**
Average	54.1 %	74.7%	**77.6%**

Table 1: Test split results of development set

Overall, 3-ResNet outperforms 2-ResNet in most of the classes, though the average accuracy score is not much different.

5. Conclusion

The achieved result from 3-ResNet has higher accuracy than the one in the original paper about 3% [2], but there's no significant difference in accuracy for 2-ResNet compared to the original paper [3]. For 3-ResNet, this may be because we use data that were originally not included in the dataset. This extra dataset included data from 3 simulated devices S4 to S6 (750 segments each device) and 3855 segments from device A. Moreover, initially, segments from devices S4 to S6 were not used in the train split, but still exists in the test split of the development set. Therefore, by including them in the training phase, the model was able to learn and predict better with the test split.

However, in this project, due to time constraint, we did not perform any kinds of data augmentation which is one the best methods boosting the performance of the model by introducing a wider variety to the input data. Therefore, this helps the model generalize well on different dataset.. However, this step is very important to make predictions on the development test split and private evaluation dataset, since the recording devices are different from the ones employed in the development set.

6. References

[1] http://dcase.community/challenge2020/

[2] S. Suh, S. Park, Y. Jeong and T. Lee, “Designing Acoustic Scene Classification Models with CNN Variants,” in DCASE2020 Challenge, Tech. Rep

[3] Hu, Hu and Yang, Chao-Han Huck and Xia, Xianjun and Bai, Xue and Tang, Xin and Wang, Yajian and Niu, Shutong and Chai, Li and Li, Juanjuan and Zhu, Hongning and Bao, Feng and Zhao, Yuanjun and Siniscalchi, Sabato Marco and Wang, Yannan and Du, Jun and Lee, Chin-Hui, “Device-Robust Acoustic Scene Classification Based on Two-Stage Categorization and Data Augmentation,” DCASE2020 Challenge, Tech. Rep.

[4] https://wiki.aalto.fi/display/ITSP/Deltas+and+Delta-deltas

[5] Koutini, K., Eghbal-zadeh, H., andWidmer, G. (2019), “CP-JKU submissions to DCASE’19: Acoustic Scene Classification and Audio Tagging with Receptive-Field-Regularized CNNs,”inProceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019).

[6] Gao, W., andMcDonnell, M. (2019),“Acoustic scene classification using deep residual networks with late fusion of separated high and low frequency paths,”DCASE2019 Challenge,Tech. Rep.

[7] Phaye, S. S. R., Benetos, E., andWang, Y. (2019, May), “SubSpectralNet–using sub-spectrogram based convolutional neural networks for acoustic scene classification,”In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)(pp. 825 - 829), IEEE.