

CNN-Based Acoustic Scene Classification System

Yerin Lee [†], Soyoung Lim [†] and Il-Youp Kwak ^{*†} 

Department of Applied Statistics, Chung-Ang University, Seoul 06974, Korea; yenini0206@gmail.com (Y.L.); isy92123@gmail.com (S.L.)

* Correspondence: ikwak2@cau.ac.kr; Tel.: +82-2-820-5390

[†] These authors contributed equally to this work.

Abstract: Acoustic scene classification (ASC) categorizes an audio file based on the environment in which it has been recorded. This has long been studied in the detection and classification of acoustic scenes and events (DCASE). This presents the solution to Task 1 of the DCASE 2020 challenge submitted by the Chung-Ang University team. Task 1 addressed two challenges that ASC faces in real-world applications. One is that the audio recorded using different recording devices should be classified in general, and the other is that the model used should have low-complexity. We proposed two models to overcome the aforementioned problems. First, a more general classification model was proposed by combining the harmonic-percussive source separation (HPSS) and deltas-deltadeltas features with four different models. Second, using the same feature, depthwise separable convolution was applied to the Convolutional layer to develop a low-complexity model. Moreover, using gradient-weight class activation mapping (Grad-CAM), we investigated what part of the feature our model sees and identifies. Our proposed system ranked 9th and 7th in the competition for these two subtasks, respectively.

Keywords: convolutional neural network; voice classification; ResNet; ensemble



Citation: Lee, Y.; Lim, S.; Kwak, I.-Y. CNN Based Acoustic Scene Classification System. *Electronics* **2021**, *10*, 371. <https://doi.org/10.3390/electronics10040371>

Academic Editor: Osvaldo Gervasi
Received: 12 January 2021
Accepted: 30 January 2021
Published: 3 February 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, acoustic scene classification (ASC) has attracted widespread attention in the Audio and Acoustic Signal Processing (AASP) community [1–6]. ASC aims to classify a test recording sound into predefined classes that characterizes the environment in which it was recorded [7]. The IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE) takes place every year. It started from 2013 and is continuing every year since 2016. The current year's ASC task is divided into two subtasks: A and B.

Figure 1a shows the problem overview on subtask A. Subtask A aims to classify audio into ten classes. Related audio files are recorded or simulated using multiple devices. The development dataset includes 40 h of data from device A and smaller amounts of data from other devices. The audio is provided in a single-channel 44.1 kHz 24-bit format. Figure 1b shows the problem overview on subtask B.

Subtask B aims to classify audio into three classes based on low-complexity solutions. All participants were required to comply with the model size of 500 KB or less. The related dataset contains data recorded by a single device (device A). The audio is provided in a binaural 48 kHz 24-bit format.

Convolutional neural networks (CNNs) are deep neural networks that are commonly used for visual image analysis. CNN is applied to computer vision, natural language processing, and recommendation systems. Moreover, it can be applied to audio by transforming the audio of the waveform into an image. For audio, CNN can be used pertaining to the pre-processing of raw data, such as short time Fourier transform (STFT), constant-Q transform (CQT), and mel-frequency cepstral coefficient (MFCC). The top-performing studies in the recent ASC competitions used CNNs. Han and Park [8] learned deep learning models for left-right (LR), mid-side (MS), and harmonic-percussive source separation

(HPSS) features and ensembled these results to win second prize at the 2017 ASC competition. Sakashita and Aono [9] performed feature extraction using LR and HPSS and won the 2018 ASC competition using the model proposed by Han and Park [8]. Gao and McDonnell [10,11] used the deltas-deltadeltas feature and ranked second place in the 2019 ASC competition. In the last competition, only real and seen devices were provided, and simulated or unseen devices were not used. Moreover, there were no tasks that considered the low complexity criterion.

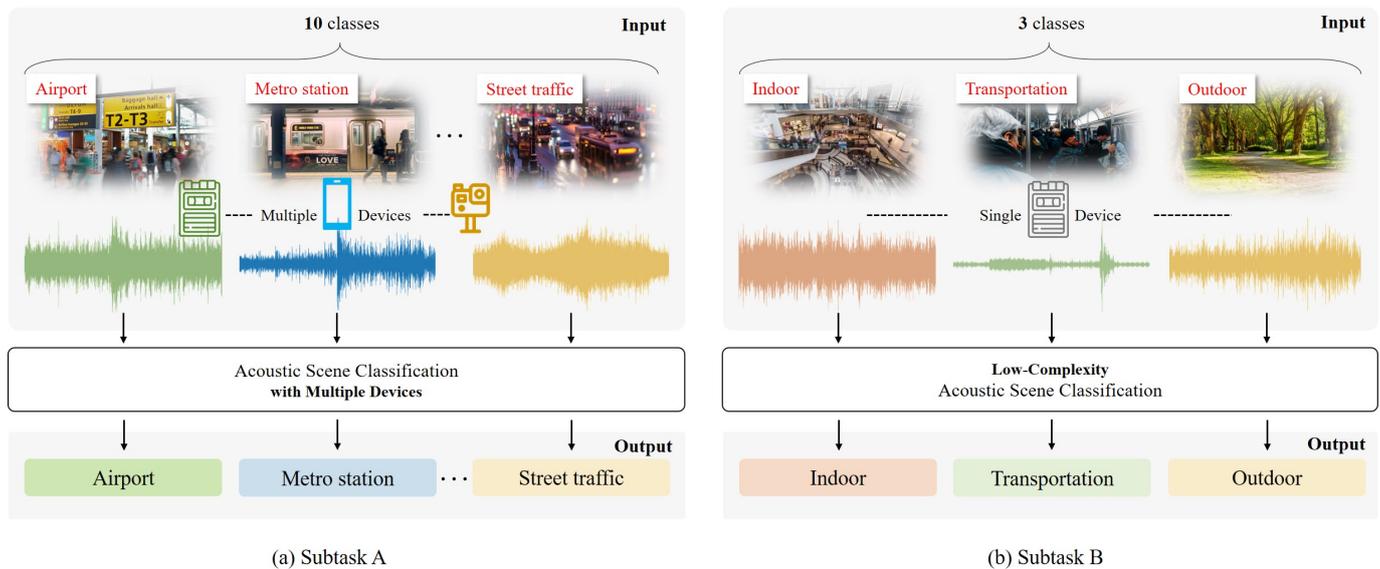


Figure 1. Overview of the ASC system for (a) subtask A and (b) subtask B.

In the field of deep learning, several studies have been conducted to derive a light-weight model. Xception proposed by Chollet [12], MobileNet proposed by Sandler et al. [13], MobileNetV2 proposed by Howard et al. [14], ANTNets proposed by Xiong et al. [15], and Shuffle Net proposed by Zhang et al. [16] have been studied to efficiently reduce computation steps for the convolution filter. Han et al. [17] proposed a pruning method that removes unnecessary parameters of the existing algorithm and a quantization method to reduce the size of the existing model without losing expressive power by reducing the parameter by a specific number of bits. Ullrich et al. [18] proposed weight sharing to have common values of parameters.

This study proposes novel mechanisms to develop voice classification systems for subtasks A and B. Our key contributions are summarized below:

- In subtask A, we modified ConvBlock in the model proposed by Han and Park [8] and Sakashita and Aono [9] to consider complex environments (multiple devices). In addition to the HPSS feature, the deltas-deltadeltas feature proposed by Gao and McDonnell [10,11] was used. In the existing model, ConvBlock was modified to reflect the concepts of ResNet, InceptionNet, and LCNN.
- In subtask B, we reduced the number of ResNet layers, and the model size was reduced by using the depth-wise separable convolution used in MobileNet [14].
- Gradient-weight class activation mapping(GradCAM) was used to determine the parts of the feature for each environment that led to the final classification decision. The characteristics of each environment were examined, based on which the possibility of interpreting the model was confirmed.

2. Materials and Methods

The speech classification model is divided into two parts: voice features and modeling. Section 2.1 introduces audio preprocessing, which is used equally for subtasks A and B, and Section 2.2 defines a convolutional network model for two acoustic scene classification tasks.

2.1. Audio Preprocessing

The audio preprocessing part in Figure 2 describes how we process audio data. We used a log-mel-spectrogram. The audio files in subtask A are monaural and have a sampling rate of 44.1 kHz. The audio files in Subtask B are binaural and have a sampling rate of 48 kHz. To generate each spectrogram, we used 2048 fast Fourier transform basis functions, a hop-length of 1024 samples, 128 frequency bins, and the hidden Markov model toolkit (HTK). Subsequently, we extracted a log-mel-spectrogram.

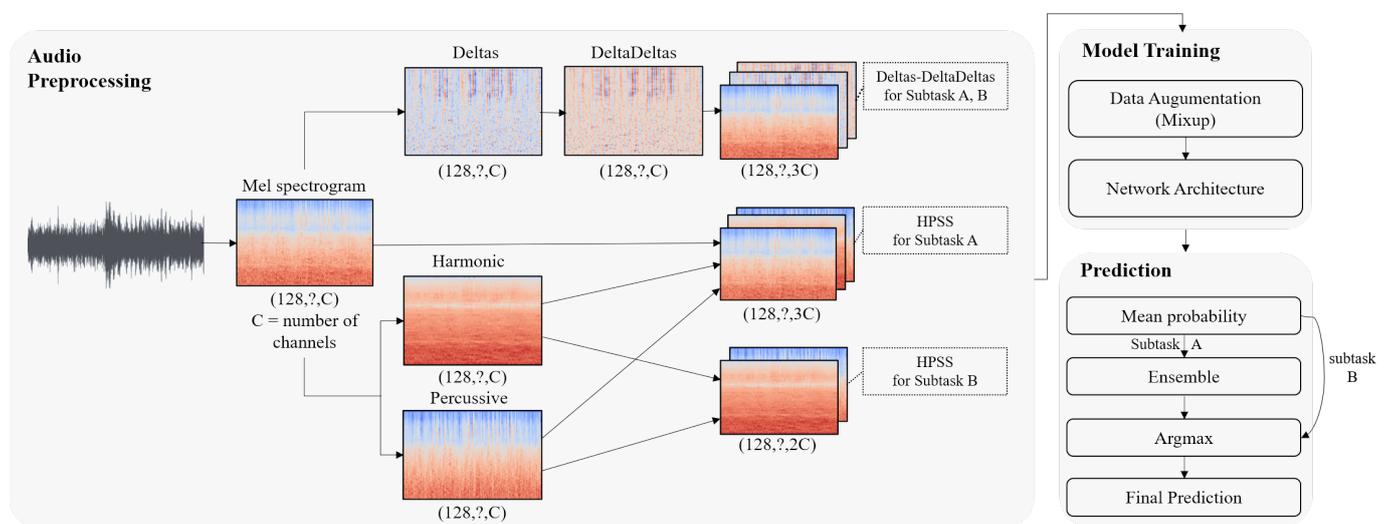


Figure 2. Proposed system architecture.

2.1.1. Harmonic-Percussive Source Separation

The HPSS decomposes monaural audio into two channels: (1) harmonic and (2) percussive. Harmonic sound is perceived as pitched sound and enables us to hear melodies and chords. In contrast, percussive sound is more related to noise and usually stems from instrument onsets, e.g., hitting on a drum. An important characteristic of percussive sounds is that they do not have pitch but have clear localization in time. In addition, the mel-spectrogram was obtained from HPSS applied to monaural audio. We used three channels (log-mel, harmonic, and percussive) for subtask A (monaural) and four channels (harmonic (left, right), percussive (left, right)) for subtask B (binaural).

2.1.2. Log-Mel-Spectrogram, Deltas, and Delta-Deltas

For feature extraction, our approach was inspired by the work of McDonnell for the ASC challenge 2019 [10,11], which used log-mel-spectrogram, deltas, and delta-deltas obtained from the log-mel-spectrogram. The deltas and delta-deltas indicate the first and second temporal derivatives of the spectrogram, respectively. For subtask A, with monaural samples in raw data, we used three input channels (log-mel, deltas, and delta-deltas) for our deep learning models. For subtask B, we used six input channels (as we have binaural samples).

2.1.3. Data Augmentation

Mixup is an effective data augmentation method [19]. We used a general augmentation approach: we mixed different samples of the training set according to their weights. The method is as follows:

$$X = \lambda X_i + (1 - \lambda) X_j \quad (1)$$

$$y = \lambda y_i + (1 - \lambda) y_j \quad (2)$$

where $\lambda \in [0, 1]$ and is acquired by the sampling of the beta distribution with parameter α , $\beta(\alpha, \alpha)$, $\alpha \in (0, \infty)$. X_i and X_j are different data samples; y_i and y_j are their corresponding labels. In our experiment, we used the mixup to augment the log-mel-spectrograms. We set α at 0.4 and used crop augmentation as 400 on the temporal axis before the mixup augmentation [10].

2.2. Network Architecture

Most top teams of 2019 used CNNs for audio data. We applied CNNs on preprocessed features (log-mel-spectrogram, deltas, delta-deltas, and HPSS). Figure 2 illustrates our architecture. Subtasks A and B are similar; however, we avoid ensembles in subtask B to maintain a lightweight model. In the following sections, we describe the network architectures for subtasks A and B.

2.2.1. Benchmarking Model for Subtask A

The model developed by Han and Park [8] ranked second in ASC challenge 2017. This model ensembles four models using different inputs. The first model uses a 1-channel mel-spectrogram with mono audio input. The model architecture (a) in Figure 3 and CNN ConvBlock (b) were used as learning models. That is, the received feature passes through a model comprising of four ConvBlocks and two fully connected layers. In ConvBlock, the padding layer, batch normalization layer, activation layer, and convolutional layer were repeated twice. The activation layers use rectified linear unit (ReLU) activation and the convolutional layers use 3×3 filters. The number of filters of the convolutional layers of the four ConvBlocks initially 32 and doubles up to 256. Each ConvBlock is followed by a max pooling layer, which is followed by a global average pooling layer after the last ConvBlock. The first fully connected layer has 1024 units and uses ReLU as an activation function. The last fully connected layer has 10 units (the number of classes) and uses the Softmax activation function. The second model uses a 2-channel mel-spectrogram in which the audio received in mono is separated by HPSS. They train the first model, but concatenate the two features received before the fully-connected layer. The other two models use a 2-channel mel-spectrogram separated by left-right (LR) or mid-side (MS) by receiving audio as stereo, and they use the same learning model as the second model based of the HPSS feature. Han and Park [8] ensembled the four models learned as described above.

Sakashita and Aono [9] won ASC challenge 2018 by ensembling a model, as shown in Figure 3a with b as its ConvBlock similar to the work of Han and Park [8]. The difference is that Sakashita and Aono [9] used nine diverse features.

After confirming that the architecture (Figure 3a) performed well in the 2017 and 2018 competitions, we used it as a baseline model.

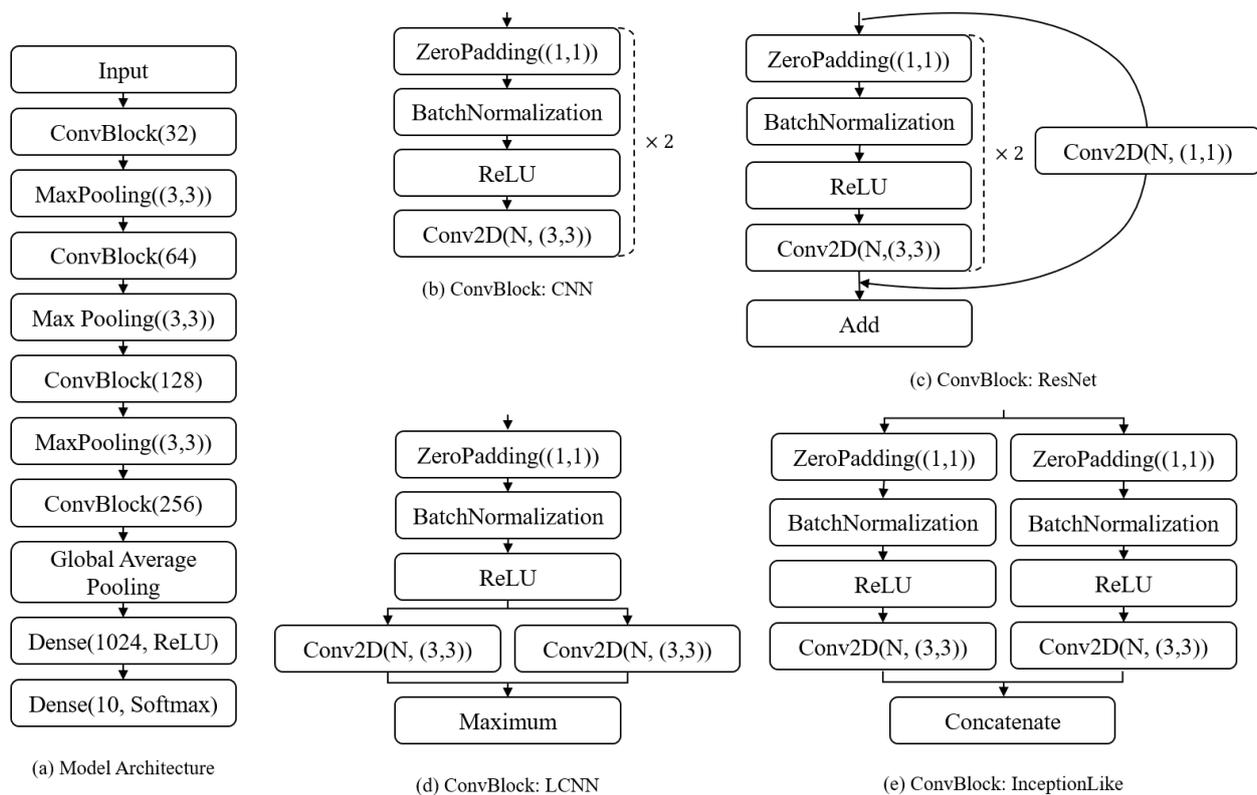


Figure 3. CNN architecture for subtask A. (a) General model architecture. (b–e): Four different modeling architectures for ConvBlock inspired by VGGNet, ResNet, LCNN, and InceptionNet, respectively.

2.2.2. Proposed Ensemble Model for Subtask A

We modified ConvBlock to consider more complex environments (multiple devices) in the existing models proposed by Han and Park [8] and Sakashita and Aono [9]. While maintaining the architecture of the existing model, the ConvBlock was transformed by considering ResNet, InceptionNet, and LCNN. In addition, for the input feature, we additionally considered the deltas-deltadeltas feature proposed by Gao and McDonnell [10] and HPSS.

Figure 3 depicts our CNN architecture for subtask A: (a) shows the general model architecture, and we proposed to use four different architectures (b)–(e) for the ConvBlock. The first architecture for ConvBlock (b) is the CNN module developed by Sakashita and Aono [9] and Han and Park [8]. This neural network is a convolutional model based on VGGNet [20]. The second architecture, (c), uses the skip-connection network in ResNet [21]. The third one, (d), is an LCNN module that uses Max-Feature-Map (MFM) activation inside the skip-connection network [22]. The fourth one, (e), is based on InceptionNet that concatenates two tensors [23].

2.2.3. Proposed Model for Subtask B

Figure 4 depicts our architecture for subtask B. The proposed reduced ResNet architecture is illustrated in (d); HPSS and deltas-deltadeltas are used as inputs for this model. We have seven convolutional layers and one shortcut connection in our reduced ResNet. In addition, the convolutional layer was replaced with a depthwise separable convolution layer for additional model weight reduction.

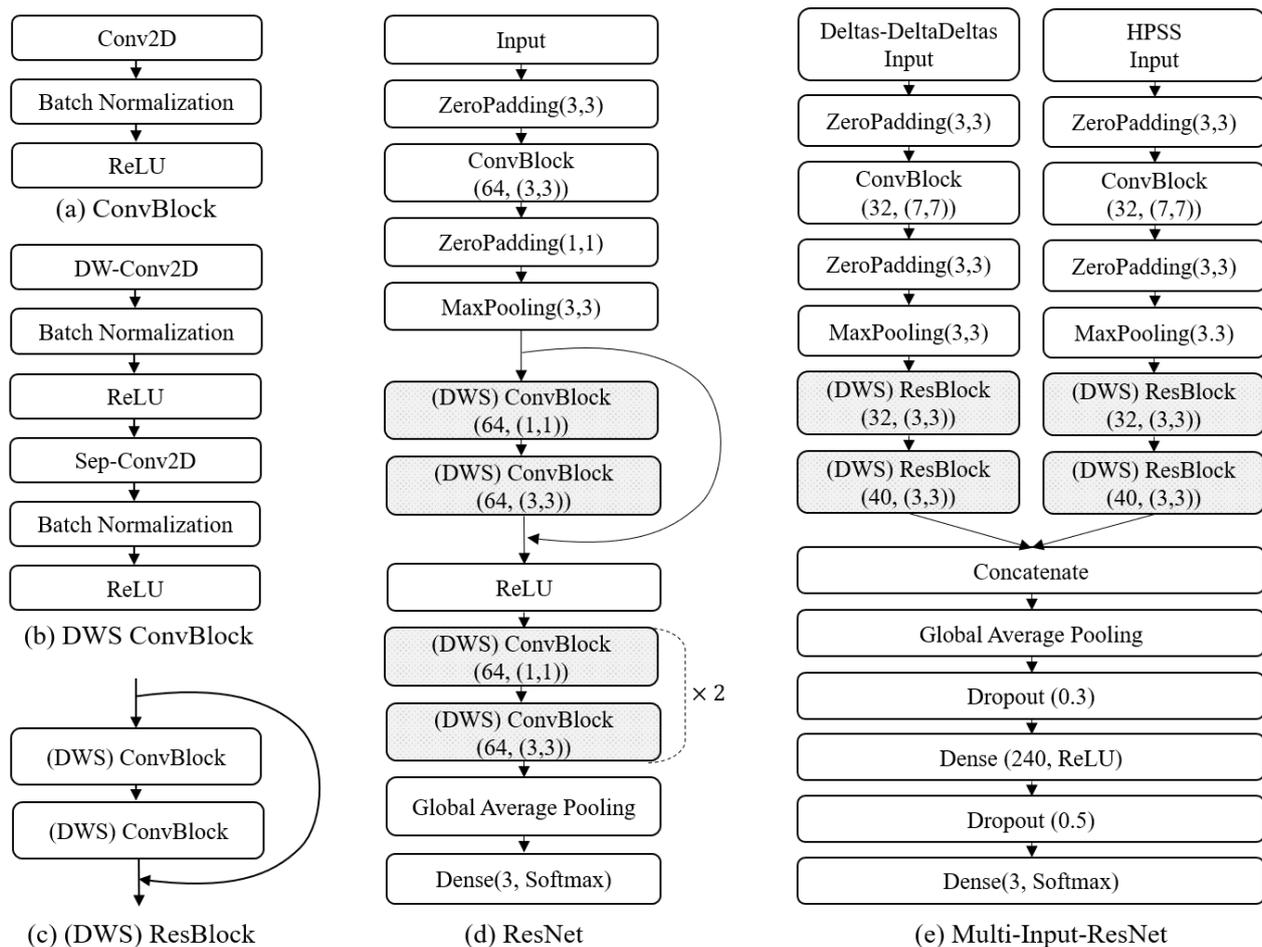


Figure 4. Architecture for subtask B. (a) and (b) describe submodules ConvBlock and depthwise separable ConvBlock. (c) describe ResBlock. (d) Proposed reduced ResNet model. (e) Proposed multi-input ResNet model.

ConvBlock used in (d) is (a) comprises a convolutional layer, batch normalization, and activation function. ReLU was used as the activation function. To make the model light, global averaging pooling was used instead of the flattened layer, and the fully connected layer was not used.

Moreover, we considered the multi-input model shown in (e); two input features are HPSS and deltas-deltadeltas. Each input had its own convolutional layers to extract the features, which were then concatenated and fed to the subsequent fully connected layers. First, the HPSS and deltas-deltadeltas features are trained through each layer composed of ConvBlock (a) and ResBlock (c), and after the concatenation of the two layers, they pass through the Global Average Pooling layer. The fully connected layer was not used for model efficiency.

Depthwise separable (DWS) convolution has been proposed in Xception and MobileNet papers [12,14]. The filter of the existing convolution layer is divided into depthwise and pointwise convolution parts. Depthwise convolution learns a filter for each channel separately and returns the output of all the channels. A pointwise convolution is applied to this output.

In the case of applying DWS in [14], there was a gain of eight to nine times the operation speed, but the performance slightly degraded. As shown in Figure 4d,e, we examined the performance of DWS, the version with DWS, applied to the (DWS) ConvBlock and the one without. In the model with DWS, DWS ConvBlock (b) is used for (DWS) ConvBlock instead of ConvBlock (a).

3. Experiments

3.1. Datasets

All datasets were collected by the Tampere University of Technology (TAU) between May and November 2018 [7].

Subtask A used the TAU Urban Acoustic Scenes 2020 Mobile, Development dataset. The dataset contains recordings obtained from 10 European cities using 9 different devices: 3 real devices (A, B, and C) and 6 simulated devices. Data from devices B, C, and simulated devices consists of randomly selected segments of the simultaneous recordings; therefore, all the data overlap with the data of device A but not necessarily with each other.

The dataset is provided with a training/test split in which 70% of the data for each device is included for training and 30% for testing. Some simulated devices appear only in the test subset.

The dataset for subtask B is the TAU Urban Acoustic Scenes 2020 3Class, Development dataset. It contains recordings obtained from 10 European cities in 10 different acoustic scenes using a single device (device A). The scenes are grouped into three major classes: indoor, outdoor, and transportation.

The dataset is provided with a training/test split in which 70% of the data is included for training and 30% for testing.

3.2. Models

For subtasks A and B, we used various model configurations. The naming conventions are as follows:

- “HPSS”: HPSS features described in Section 2.1.1.
- “Deltas-DeltaDeltas”: log-mel energies, deltas, and delta-deltas features described in Section 2.1.2.
- “CNN”: the CNN architecture from Figure 3b.
- “ResNet”: the ResNet architecture from Figure 3c for subtask A and Figure 4 for subtask B.
- “DWS-ResNet”: the ResNet architecture with a DWS convolution layer from Figure 4 for subtask B.
- “LCNN”: the LCNN architecture from Figure 3d.
- “InceptionLike”: the Inception-like architecture from Figure 3e.
- “Deltas-DeltaDeltas-Ensemble”: This model ensembled four “Deltas-DeltaDeltas” models.
- “HPSS-Ensemble”: This model ensembled four “HPSS” models.
- “All-Ensemble”: An ensemble of all 8 models with equal weights.
- “Multi-Input”: both Deltas-DeltaDeltas and HPSS input features for subtask B.

We trained all our models in 100 epochs. After 60 epochs, the learning rate was reduced from 10^{-3} to 10^{-5} . We used the Adam optimizer and applied a sigmoidal decay using a learning rate scheduler implemented in Keras.

In subtask A, we used an ensemble to improve the performance (because different efficient models trained independently are likely to be effective for different reasons). We computed initial predictions using each separate model. Subsequently, we used the average and argmax functions for prediction.

3.3. Grad-CAM Visualization

If the proposed model has the ability to explain the reason for making predictions, i.e., the ability to interpret, then it will help when the model is ineffective and when it fits well. When the model is ineffective, the reason for failure can be determined, and when it fits well, its reliability can be determined. Several techniques have been developed to visualize and interpret learning expressions. Among these, we use a technique called Gradient-weight Class Activation Mapping (Grad-CAM) to visualize the parts of the feature based on which the model judges [24].

Grad-CAM can help understand the parts of the image that lead the CNN to make the final classification decision. Let us predict a picture as class A. Considering the specific convolution layer of our model, the predicted value is a function of specific convolution layer values. The model is trained to minimize the loss between the predicted value and class A; a heatmap can be drawn by applying gradient accents to the direction of increasing loss and summing the absolute changes in specific convolution layer values along the channel direction.

As Grad-CAM calculates weight using a gradient, it can be applied to any layer. As it is known to read semantic class-specific information as it goes deeper in the CNN, we will see the result of Grad-CAM using the last layer of our model. We applied Grad-CAM to the Deltas-DeltaDeltas-Resnet model and the Deltas-DeltaDeltas-DWS model, which had the best performance on subtask B.

4. Results

This section reports the performance of our proposed models on the development set for subtasks A and B.

4.1. Evaluation Metric

We compared and evaluated the performance of our proposed approaches for subtasks A and B. As evaluation metrics, we used the multi-class accuracy, macro-averaged recall, and macro-averaged-precision. The multi-class accuracy can be obtained as the ratio of correct predictions to the number of test examples.

$$Accuracy = \frac{1}{N} \sum_{i=1}^N I(y_i = \hat{y}_i) \quad (3)$$

where I is an indicator function and returns 1 if the classes match, and 0 otherwise.

Macro-averaged recall is the average of the class-wise recalls, and macro-average precision is the average of the class-wise precisions. If the number of predefined classes is C , the macro-averaged recall R_{macro} and the macro-averaged precision P_{macro} can be calculated as follows:

$$R_{macro} = \frac{1}{C} \sum_{i=1}^C \frac{TP_i}{TP_i + FP_i} = \frac{\sum_{i=1}^C R_i}{C} \quad (4)$$

$$P_{macro} = \frac{1}{C} \sum_{i=1}^C \frac{TP_i}{TP_i + FN_i} = \frac{\sum_{i=1}^C P_i}{C} \quad (5)$$

where TP_i , FP_i , and FN_i represent true positives, false positives, and false negatives in the i th class, respectively, and R_i and P_i represent recall and precision in the i th class.

4.2. Classification Results for Subtask A

We used HPSS-CNN proposed by Han and Park [8] and Sakashita and Aono [9] as the baseline model and compared it with our proposed models. Table 1 lists the experimental results for subtask A. Each model applied to Deltas-DeltaDeltas in subtask A had an accuracy of over 60%, while the same model applied to HPSS had a lower accuracy. When HPSS was used as a feature, when comparing the accuracy of the four models CNN, ResNet, LCNN, and InceptionLike, the accuracy of the CNN proposed in Han and Park [8] and Sakashita and Aono [9] was the highest at 59.36%. On the other hand, with Deltas-DeltaDeltas, the accuracy of CNN was the lowest at 63.43% among the four models, and the accuracy of ResNet was the highest at 64.21%. The accuracy of Deltas-DeltaDeltas-Ensemble is 69.16%, which is 5.12% higher than that of the HPSS-Ensemble. In addition, it has higher accuracy than the All-Ensemble that ensembles all eight models.

Table 1. Subtask A Results. Accuracy (%), recall (%) and precision (%) are multi-class accuracy, macro-averaged recall, and macro-average precision, respectively, of the development set.

Model	Model Configuration (Subtask A)	Accuracy (%)	Recall (%)	Precision (%)
1	HPSS-CNN	59.36	59.36	60.27
2	HPSS-ResNet	58.55	58.55	59.93
3	HPSS-LCNN	58.08	58.08	58.35
4	HPSS-InceptionLike	58.69	58.69	58.97
5	Deltas-DeltaDeltas-CNN	63.43	64.43	63.60
6	Deltas-DeltaDeltas-ResNet	64.21	64.21	65.23
7	Deltas-DeltaDeltas-LCNN	64.11	64.11	64.58
8	Deltas-DeltaDeltas-InceptionLike	63.94	63.94	63.94
9	HPSS-Ensemble	64.04	64.04	64.33
10	Deltas-DeltaDeltas-Ensemble	69.16	69.16	68.86
11	All-Ensemble	68.62	68.62	68.61

Table 2 lists the classification accuracy results of 10 classes. With Deltas-DeltaDeltas-ResNet, the accuracy improved for eight classes, excluding Shopping Mall and Tram, compared with HPSS-CNN, the baseline model. In particular, the accuracy significantly increased for Bus, Park, and Street Pedestrian. When Deltas-DeltaDeltas-Ensemble was used, the accuracy for the Tram significantly increased compared to when the single Deltas-DeltaDeltas-ResNet model was used.

Table 2. Subtask A class-wise performance. HPSS-CNN (%), Deltas-DeltaDeltas-ResNet (%), and Deltas-DeltaDeltas-Ensemble (%) are class-wise accuracy (recall) of the development set.

Classes	HPSS-CNN (%)	Deltas-DeltaDeltas-ResNet (%)	Deltas-DeltaDeltas-Ensemble (%)
Airport	48.48	50.17	56.57
Bus	62.96	78.45	80.47
Metro	57.58	64.31	69.70
Metro Station	62.63	68.01	69.02
Park	69.36	80.13	85.19
Public Square	46.13	45.45	52.86
Shopping Mall	66.00	65.32	67.34
Street Pedestrian	33.00	42.42	48.48
Street Traffic	78.79	81.82	84.18
Tram	68.69	66.00	77.78

Table 3 lists the classification accuracy results of nine devices. In all the devices except device A, when Deltas-DeltaDeltas-ResNet was used, the classification accuracy was higher than when HPSS-CNN, the baseline model, was used. In particular, the accuracy of the unseen devices (S4–S6), which were not used for training, relatively increased compared with the seen devices (S1–S3). Deltas-DeltaDeltas-Ensemble is more accurate than HPSS-CNN and Deltas-DeltaDeltas-ResNet for all devices.

Table 3. Subtask A device-wise performance. Devices A, B, C, and S1–S3 are seen devices, and S4–S6 are unseen devices. HPSS-CNN (%), Deltas-DeltaDeltas-ResNet (%), and Deltas-DeltaDeltas-Ensemble (%) are class-wise accuracy of the development set.

Devices	HPSS-CNN (%)	Deltas-DeltaDeltas-ResNet (%)	Deltas-DeltaDeltas-Ensemble (%)
A	74.45	72.42	80.00
B	64.24	68.79	73.03
C	66.67	69.39	76.67
S1	62.42	64.85	70.90
S2	53.94	58.79	63.03
S3	59.70	61.62	70.60
S4	52.42	58.18	64.24
S5	53.64	56.06	62.42
S6	46.67	56.06	61.51

4.3. Classification Results for Subtask B

Table 4 and Figure 5 present the experimental results for subtask B. Deltas-DeltaDeltas-ResNet, HPSS-ResNet, and Multi-Input-ResNet in Table 4 indicate the accuracy and model size of reduced ResNet and multi-input models without DWS in the convolution layer. The ResNet model performance using the Deltas-DeltaDeltas feature was the highest at 95.38%. As the model weight reduction technique was not used, the model size is smaller than the 500 KB limit in the competition, but its size is close to the limit. Deltas-DeltaDeltas-DWS-ResNet, HPSS-DWS-ResNet and Multi-Input-DWS-ResNet indicate the results of the models based on the model weight reduction technique (DWS convolution). Overall, the performance of the models with DWS has slightly decreased, but we can see that the model size is more than three times smaller with DWS convolutions.

Table 4. Subtask B Results. Accuracy (%), recall (%), and precision (%) are multi-class accuracy, macro-averaged recall and macro-average precision, respectively, of the development set.

Model	Model Configuration (Subtask B)	Accuracy (%)	Recall (%)	Precision (%)	Size (KB)
1	Deltas-DeltaDeltas-ResNet	95.38	95.32	95.25	494.2
2	HPSS-ResNet	95.29	95.33	95.19	489.8
3	Multi-Input-ResNet	94.57	94.81	94.38	495.6
4	Deltas-DeltaDeltas-DWS-ResNet	95.38	95.30	95.32	131.8
5	HPSS-DWS-ResNet	94.07	94.00	93.94	127.2
6	Multi-Input-DWS-ResNet	92.90	92.94	92.77	204.3

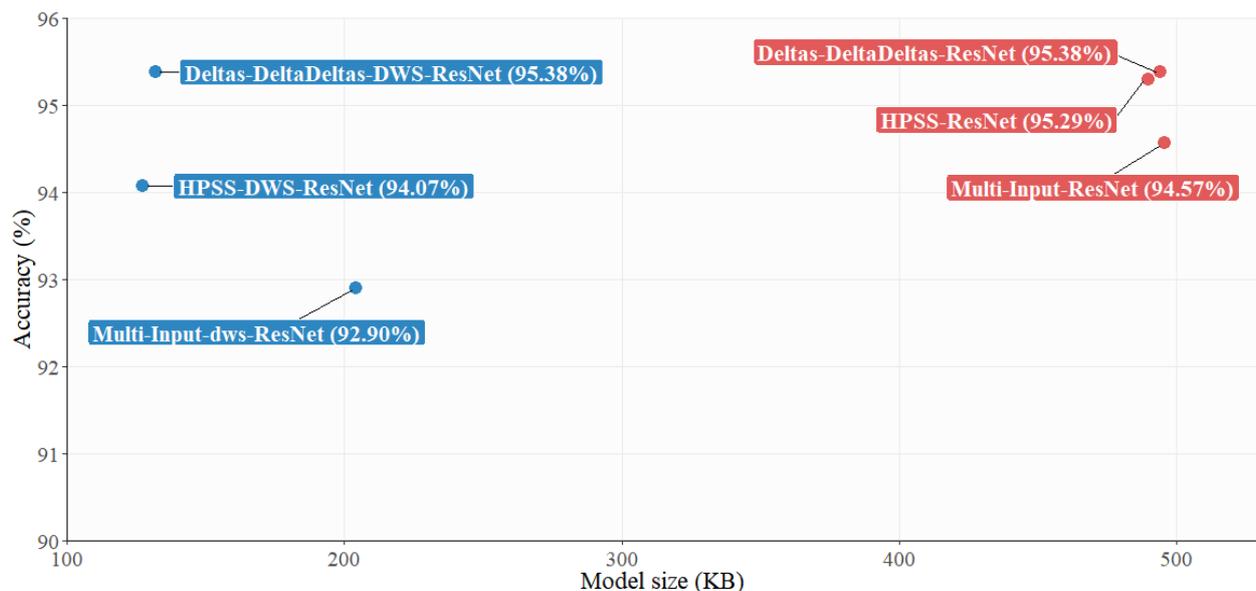


Figure 5. Graph comparing the model size and performance of subtask B. The blue dots are the models to which the weight reduction technique is applied, and the red dots are the original models.

4.4. Grad-CAM Application

Subtask B has three large labels, and there are ten place labels, which are defined in detail. Figure 6 presents the wave plot, spectrogram, and Grad-CAM results of representative voices.

Based on the corresponding voices and Grad-CAM results, the characteristics of each environment were summarized as follows.

- Indoors involve considerable human noise. The Grad-CAM results indicate that there were many activation spots.

- A characteristic feature of the Grad-CAM results of transportation is that the vertical lines exhibit a pattern in the middle. While listening to the actual recording, vertical lines appear when a rattling sound is heard.
- The distinctive feature of the Grad-CAM results of outdoor is that they have long horizontal lines. We hear continuous sound, i.e., the sound of water, walking, and birds.

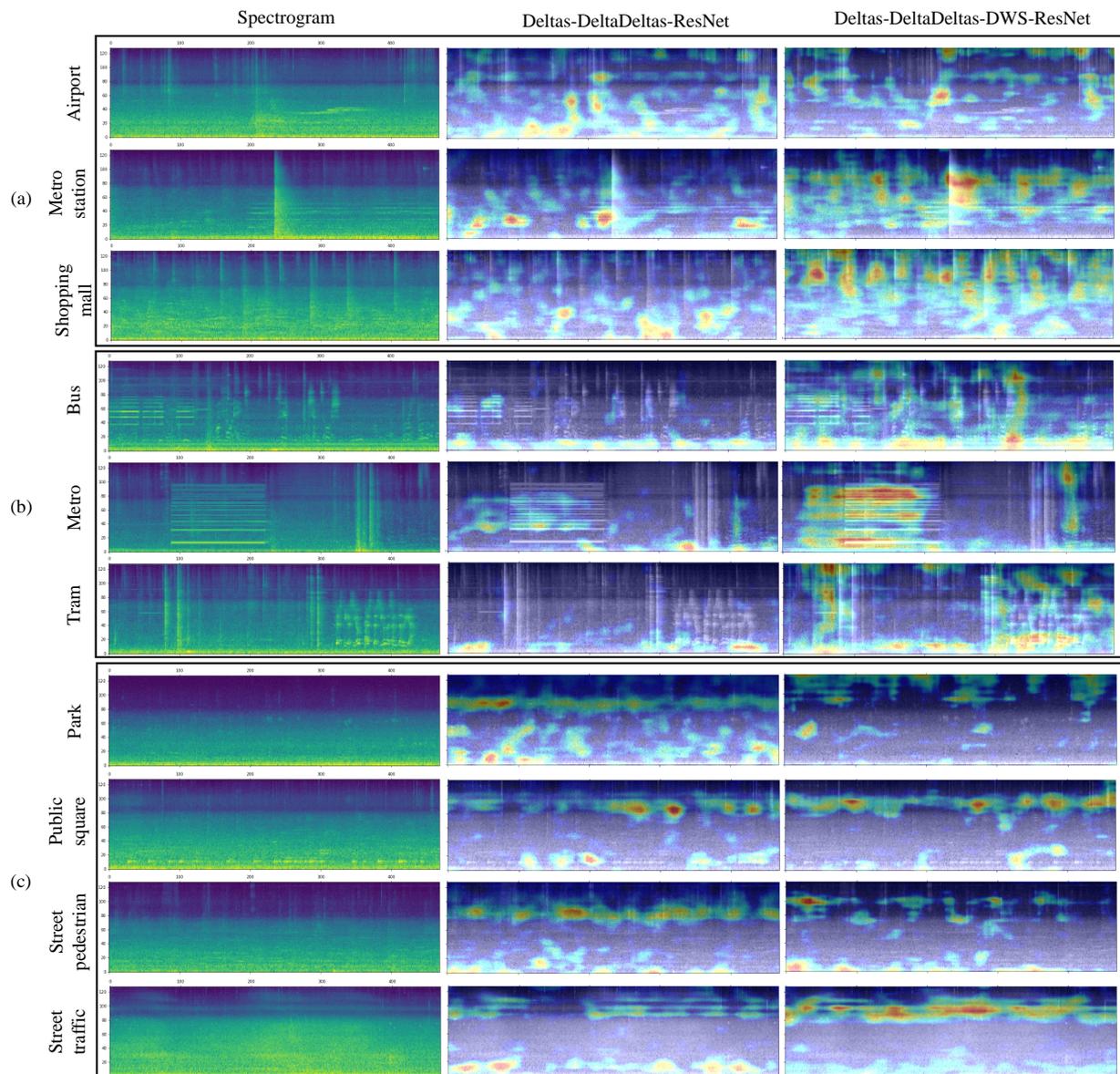


Figure 6. Grad-CAM results: (a) indoor, (b) transportation, and (c) outdoor

4.5. Misclassified Samples

Figure 7 shows the class-wise performance of subtask A and subtask B as a normalized confusion matrix. Here, each row represents a predicted class for a true class.

Figure 7a is the confusion matrix for HPSS-CNN, Deltas-DeltaDeltas-ResNet, and Deltas-DeltaDeltas-Ensemble of Table 2 for subtask A. When Deltas-DeltaDeltas-ResNet was used, the problem of misclassifying Bus as Tram was reduced compared with the case of HPSS-CNN, and the accuracy in the case of Bus significantly improved; the problem of misclassifying Park as Public Square or Tram was also reduced. In addition, as a result of using Deltas-DeltaDeltas-Ensemble, it was possible to reduce the problem of misclassifying

Public Square as Street Traffic or misclassifying Street Pedestrian as Public Square, which was not resolved in Deltas-DeltaDeltas-ResNet.

Figure 7b shows a confusion matrix that exhibits class-wise performance for Deltas-DeltaDeltas-ResNet, HPSS-ResNet, and Multi-Input-ResNet listed in Table 4 for subtask b, and Figure 7c exhibits Deltas-DeltaDeltas-DWS-ResNet, HPSS-DWS-ResNet and Multi-Input-DWS-ResNet. In all the models, transportation is well classified. In particular, there were few cases of misclassifying outdoor and transportation. Overall, indoor and outdoor are often misclassified. As shown in Figure 7c, the ratio of indoor misclassification was high.

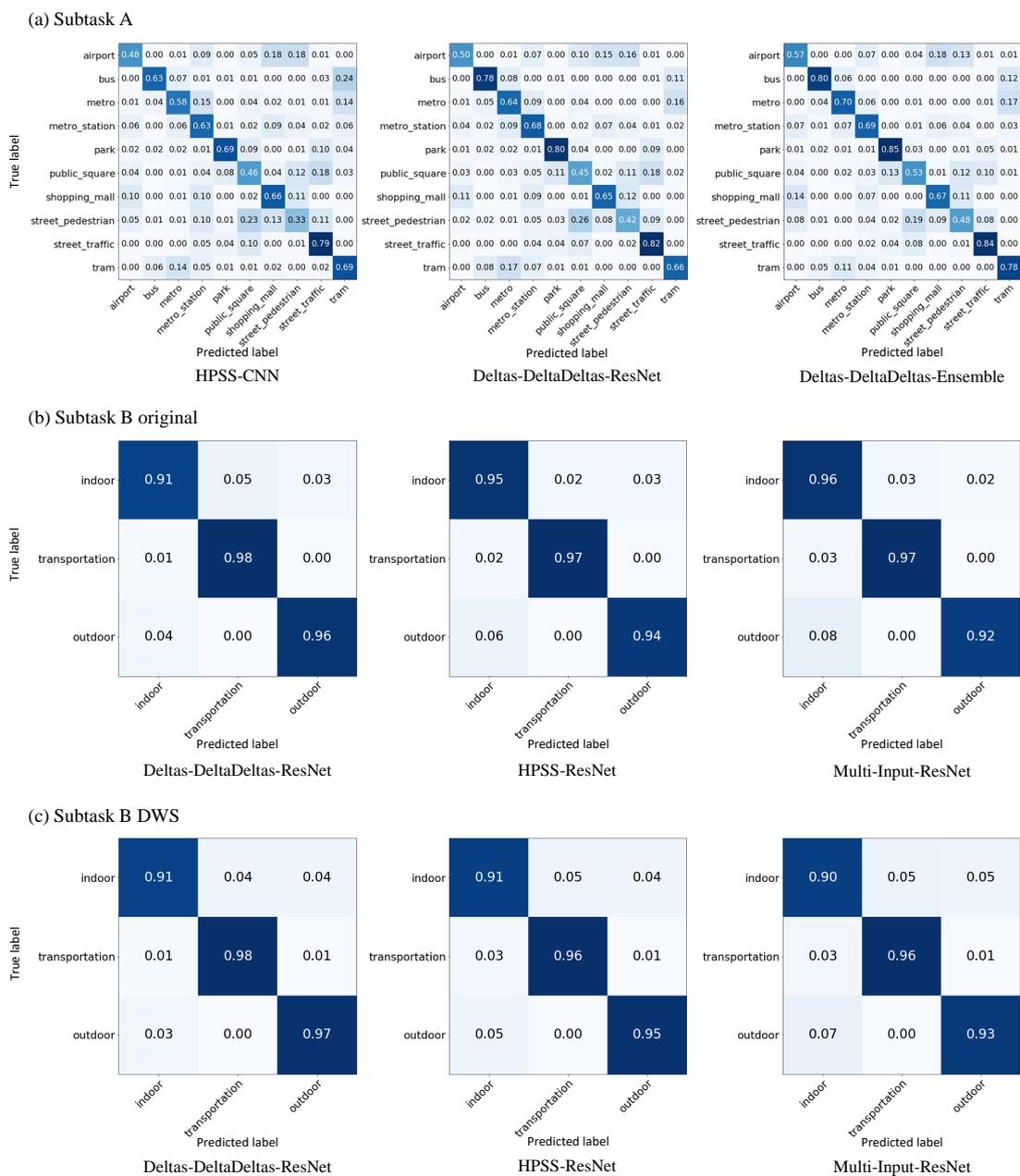


Figure 7. Class-wise performance on subtask A and B. (a) shows class-wise performance for each model of subtask A, (b) is class-wise performance for models in subtask B to which DWS is not applied, (c) is class-wise performance of models to which DWS is applied in subtask B.

5. Discussion

5.1. Comparison with Other Methods in 2020 ASC

In the 2020 ASC competition, subtask A had 92 submitted systems from 27 teams, and subtask B had 86 submitted systems from 24 teams. In the competition, evaluation was made with an undisclosed evaluation set, and macro-averaged recall was used as an evaluation index.

Among the top 10 systems of subtask A, 9 systems used the ensemble technique, and in addition to deltas-deltadeltas, various input features such as CQT, Gammatone, HPSS, and MFCC were used. All top 10 systems partially used residual network in their model architecture.

The macro-averaged recall of the best model proposed by Suh et al. [25] was 76.5%, which exceeded the macro-averaged recall of our proposed model (72.9%) by 3.6%. The top system also used deltas-deltadeltas as the input feature and combined the ideas of ResNet and concatenated multiple layers to create a more powerful model. It also used a snapshot ensemble to store the weights of all epochs as checkpoints during the training process and then the final parameter estimated by weighted averaging.

The top 10 systems of subtask B used slim models, depth-wise separable CNNs, pruning and post-training quantization to create the light-weight model. The macro-averaged recall of the top performing system proposed by Koutini et al. [26] was 96.5%, which is 2.6% higher than of the macro-averaged recall of our proposed model (i.e., 93.9%). Perceptually-weighted log-mel energies were used as input features, and they considered residual network with receptive-field-regularized CNN. Parameter pruning and quantization were used to reduce the weight of the model. Through pruning, the 6671.5 KB model was reduced to 483.5 KB.

5.2. ASC Applications

ASC can be used to a smartphone that automatically changes to silent mode or adjusts the volume based on the location, or to a hearing aid that adjusts its function according to indoor or outdoor recognition. In addition, it can be performed as a pre-processing step to address other problems such as separating the source of the speech signal from background noise [27].

Determining a place with a deep learning model learned by various devices of ASC subtask A is a task that can be applied to various products, such as smartphones, TVs, and refrigerators. Subtask B of making light-weight models is an important problem for implementing models in various low specification products.

5.3. Visualizing What Convnets Learn

One of the problems of deep learning is that the model cannot explain the reason for the submitted result, and studies have been conducted to address problem [28,29]. In the ASC problem, there have been attempts to explain the results as well [30,31].

Ren et al. [30] visualized the attention layer of the model to explain the result. Wu and Lee [31] also visualized the Grad-CAM results for metro stations, residential areas, and trains for the 2017 ASC competition data. As shown in Figure 6, the visualization results for train and metro stations were similar to those of other studies. We additionally found in our study that the background sound of an outdoor environment can be observed as a horizontal line of fixed frequency.

5.4. Limitations and Future Work

Our model won 7th and 9th ranks at the ASC competition. However, better models are also available, and our models have several shortcomings. For future research, in subtask A, various augmentation techniques such as pitch shift, speed change, random noise, and mix audios can be applied to improve accuracy [32]. In addition, performance can be improved by fine tuning using pretrained weights from an external dataset [33]. If snapshot ensemble is used, accuracy can be improved while maintaining the size of the model [25]. Thus, it

can be used for both subtask A and B. For subtask B, the invertible residual and bottleneck of MobileNetv2 [13] can be considered, or by using the concept of ANTNets [15] that additionally consider squeeze and excitation networks in the invertible residual block. We can also consider pruning and decomposed convolution for more light-weight models. Additionally, we can use self-supervised learning techniques for feature representation learning [34,35] followed by light-weighted traditional machine learning methods. In this scheme, we use deep neural networks on a large amount of sound data to learn feature representations that characterize sound data very well. For the light-weight model we can apply lighter traditional machine learning methods such as k-NN, logistic regression, support vector machine and random forest. [36–38]

6. Conclusions

In the 2020 ASC competition, the discrimination of voices recorded or simulated on multiple devices was addressed in subtask A, and a lightweight and efficient model was addressed in subtask B. Our proposed system ranked 9th and 7th in the competition for subtasks A and B, respectively.

This paper presents our novel architectures for subtasks A and B. In subtask A, we considered two features, Deltas-DeltaDeltas and HPSS, and four models inspired by VGGNet, ResNet, LCNN, and InceptionNet. In all the four models, the use of Deltas-DeltaDeltas surpassed the performance of HPSS, and among them, the use of ResNet exhibited the highest accuracy. In subtask B, reduced ResNet using the Deltas-DeltaDeltas feature exhibited the best performance at 95.38%. In addition, the reduced ResNet model applying the DWS convolution and considering low-complexity was similar or slightly inferior to the performance of the model without DWS convolution, but the model size was reduced from 494.2 to 131.8 KB. Therefore, we confirmed that the DWS convolution is effective in reducing the model size while maintaining the performance of the model.

In addition, features that specify indoor, transportation, and outdoor environments were determined and visualized using Grad-CAM. In an indoor environment, the model determines the partially generated noises, whereas in an outdoor environment, the pattern of fixed frequency background sound continues. Further, there are rattle sounds in transportation, and this generates vertical lines with small intervals.

Author Contributions: Conceptualization, Y.L., S.L. and I.-Y.K.; methodology, Y.L., S.L. and I.-Y.K.; software, Y.L. and S.L.; validation, Y.L. and S.L.; formal analysis, Y.L., S.L. and I.-Y.K.; writing—original draft preparation, Y.L. and S.L.; writing—review and editing, I.-Y.K.; visualization, Y.L. and S.L.; supervision, I.-Y.K.; funding acquisition, I.-Y.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Ministry of Science and ICT (2020R1C1C1A01013020)

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: <http://dcase.community/challenge2020/task-acoustic-scene-classification>.

Acknowledgments: The authors thank the DCASE community for hosting interesting challenges every year, as well as the anonymous reviewers whose comments greatly helped improve this manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, L.; Shi, Z.; Han, J. Pyramidal Temporal Pooling With Discriminative Mapping for Audio Classification. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2020**, *28*, 770–784. [CrossRef]
2. Jung, J.W.; Heo, H.S.; Shim, H.J.; Yu, H.J. Knowledge Distillation in Acoustic Scene Classification. *IEEE Access* **2020**, *8*, 166870–166879. [CrossRef]
3. Zhang, T.; Wu, J. Constrained Learned Feature Extraction for Acoustic Scene Classification. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2019**, *27*, 1216–1228. [CrossRef]

4. Basbug, A.M.; Sert, M. Acoustic Scene Classification Using Spatial Pyramid Pooling with Convolutional Neural Networks. In Proceedings of the 2019 IEEE 13th International Conference on Semantic Computing (ICSC), Newport Beach, CA, USA, 30 January–1 February 2019; pp. 128–131.
5. Gao, L.; Mi, H.; Zhu, B.; Feng, D.; Li, Y.; Peng, Y. An Adversarial Feature Distillation Method for Audio Classification. *IEEE Access* **2019**, *7*, 105319–105330. [\[CrossRef\]](#)
6. Abeber, J. A Review of Deep Learning Based Methods for Acoustic Scene Classification. *Appl. Sci.* **2020**, *10*, 2020.
7. Available online: <http://dcase.community/challenge2020/task-acoustic-scene-classification> (accessed on 2 January 2021).
8. Han, Y.; Park, J. *Convolutional Neural Networks with Binaural Representations and Background Subtraction for Acoustic Scene Classification*; Technical Report, DCASE2017 Challenge; DCASE Community: Washington, DC, USA, 2017.
9. Sakashita, Y.; Aono, M. *Acoustic Scene Classification by Ensemble of Spectrograms Based on Adaptive Temporal Divisions*; Technical Report, DCASE2018 Challenge; DCASE Community: Washington, DC, USA, 2018.
10. Gao, W.; McDonnell, M. *Acoustic Scene Classification Using Deep Residual Networks with Late Fusion of Separated High and Low Frequency Paths*; Technical Report, DCASE2019 Challenge; DCASE Community: Washington, DC, USA, 2019.
11. McDonnell, M.D.; Gao, W. Acoustic Scene Classification Using Deep Residual Networks with Late Fusion of Separated High and Low Frequency Paths. In Proceedings of the ICASSP 2020—2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020; pp. 141–145.
12. Chollet, F. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1251–1258.
13. Sandler, M.; Howard, A.G.; Zhu, M.; Zhmoginov, A.; Chen, L. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
14. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861,
15. Xiong, Y.; Kim, H.J.; Hedau, V. ANTNets: Mobile Convolutional Neural Networks for Resource Efficient Image Classification. *arXiv* **2019**, arXiv:1904.03775,
16. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6848–6856.
17. Han, S.; Mao, H.; Dally, W.J. Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. In Proceedings of the 4th International Conference on Learning Representations, ICLR 2016, San Juan, PR, USA, 2–4 May 2016.
18. Ullrich, K.; Meeds, E.; Welling, M. Soft Weight-Sharing for Neural Network Compression. In Proceedings of the 5th International Conference on Learning Representations (ICLR 2017), Toulon, France, 24–26 April 2017.
19. Zhang, H.; Cisse, M.; Dauphin, Y.N.; Lopez-Paz, D. mixup: Beyond Empirical Risk Minimization. In Proceedings of the 6th International Conference on Learning Representations (ICLR 2018), Vancouver, BC, Canada, 30 April–3 May 2018.
20. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015), San Diego, CA, USA, 7–9 May 2015.
21. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
22. Wu, X.; He, R.; Sun, Z.; Tan, T. A light CNN for deep face representation with noisy labels. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 2884–2896. [\[CrossRef\]](#)
23. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9.
24. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision, Honolulu, HI, USA, 21–26 July 2017; pp. 618–626.
25. Suh, S.; Park, S.; Jeong, Y.; Lee, T. *Designing Acoustic Scene Classification Models with CNN Variants*; Technical Report, DCASE2020 Challenge; DCASE Community: Washington, DC, USA, 2020.
26. Koutini, K.; Henkel, F.; Eghbal-Zadeh, H.; Widmer, G. *CP-JKU Submissions to DCASE'20: Low-Complexity Cross-Device Acoustic Scene Classification with RF-Regularized CNNs*; Technical Report, DCASE2020 Challenge; DCASE Community: Washington, DC, USA, 2020.
27. Barchiesi, D.; Giannoulis, D.; Stowell, D.; Plumbley, M.D. Acoustic Scene Classification: Classifying environments from the sounds they produce. *IEEE Signal Process. Mag.* **2015**, *32*, 16–34. [\[CrossRef\]](#)
28. Montavon, G.; Samek, W.; Müller, K.R. Methods for interpreting and understanding deep neural networks. *Digit. Signal Process.* **2018**, *73*, 1–15. [\[CrossRef\]](#)
29. Došilović, F.K.; Brčić, M.; Hlupić, N. Explainable artificial intelligence: A survey. In Proceedings of the 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 21–25 May 2018; pp. 0210–0215

30. Ren, Z.; Kong, Q.; Han, J.; Plumbley, M.D.; Schuller, B.W. CAA-Net: Conditional Atrous CNNs with Attention for Explainable Device-robust Acoustic Scene Classification. *IEEE Trans. Multimed.* **2020**. [[CrossRef](#)]
31. Wu, Y.; Lee, T. Enhancing Sound Texture in CNN-based Acoustic Scene Classification. In Proceedings of the ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 815–819.
32. Hu, H.; Yang, C.H.H.; Xia, X.; Bai, X.; Tang, X.; Wang, Y.; Niu, S.; Chai, L.; Li, J.; Zhu, H.; et al. *Device-Robust Acoustic Scene Classification Based on Two-Stage Categorization and Data Augmentation*; Technical Report, DCASE2020 Challenge; DCASE Community: Washington, DC, USA, 2020.
33. Wang, H.; Chong, D.; Zou, Y. *Acoustic Scene Classification with Multiple Decision Schemes*; Technical Report, DCASE2020 Challenge; DCASE Community: Washington, DC, USA, 2020.
34. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. In Proceedings of the 37th International Conference on Machine Learning, PMLR, Online, 13–18 July 2020; pp. 1597–1607.
35. Jing, L.; Tian, Y. Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**. [[CrossRef](#)]
36. Nguyen, B.P.; Tay, W.; Chui, C. Robust Biometric Recognition From Palm Depth Images for Gloved Hands. *IEEE Trans. Hum. Mach. Syst.* **2015**, *45*, 799–804. [[CrossRef](#)]
37. Chang, C.C.; Lin, C.J. LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.* **2011**, *2*, 27:1–27:27. [[CrossRef](#)]
38. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [[CrossRef](#)]