A DEVICE CLASSIFICATION-AIDED MULTI-TASK FRAMEWORK FOR LOW-COMPLEXITY ACOUSTIC SCENE CLASSIFICATION

Rohith Mars and Rohan Kumar Das

Fortemedia Singapore, Singapore
{rohithmars, rohankd}@fortemedia.com

ABSTRACT

In this paper, we investigate the use of a multi-task learning framework to address the DCASE 2022 Task 1 on Low-complexity Acoustic Scene Classification (ASC). Specifically, we employ classification of the recording devices as an additional task to improve the performance of the ASC task. Both these tasks utilize our proposed convolutional neural network with a shared layer along with strided and separable convolution operations designed to comply with the model parameter and computational constraints imposed by Task 1 organizers. We also explore the use of various data augmentation techniques to improve the generalization of the model. Evaluations on the development dataset show that our proposed ASC system consisting of 127.2k parameters with 27.5 million multiply and accumulate operations provides a significant improvement on overall log-loss and accuracy over the baseline system.

Index Terms— acoustic scene classification, convolutional neural networks, data augmentation, multi-task learning, quantization aware training.

1. INTRODUCTION

The objective of an Acoustic Scene Classification (ASC) system is to classify a given audio recording of a certain temporal duration into one of the pre-defined acoustic scenes. As a sub-field of computational auditory scene analysis (CASA) [1, 2], ASC has received much attention as an important and challenging research topic. It also finds several applications in consumer devices, which incorporates contextual-aware computing, such as smart-wearables, hearables as well as surveillance [3, 4].

The Detection and Classification of Acoustic Scenes and Events (DCASE) challenge[1], has played an important role in providing standard datasets for setting algorithmic benchmarks and to spearhead research on ASC. Over the years, the ASC task in DCASE challenge series has witnessed several modifications by introducing additional recording devices and constraints for model complexity. In DCASE 2021 Task 1, the model size of the proposed solutions were constrained to 128 kB for non-zero parameters, with no constraints set on the multiply and accumulate (MAC) operations. In addition, the length of audio clip for inference was set as 10 seconds for all the past editions.

A majority of the proposed systems for multi-device ASC task utilize a single task learning (STL) framework with log-Mel spectrogram as input feature to a convolutional neural network (CNN)-based model, which is trained to predict the acoustic scene [5, 6]. To improve the generalization of the model to unseen devices and recordings from unseen locations, many techniques including spectrum correction and frequency instance normalization were proposed [6, 7]. In addition to these techniques, various data augmentation strategies have also been employed extensively in the past to address the same [8, 9]. To meet the model size requirements, approaches such as model compression with pruning and use of knowledge distillation have also been well-explored [10, 11].

The latest DCASE 2022 Task 1 [12] is an extension of the previous editions of DCASE ASC challenges, with an objective to design an ASC system with low computation complexity and sufficiently less number of parameters such that it could be deployed for inference on an edge-device. Specifically, the maximum number of the model parameters is set to 128k (including the zero-valued ones) and the variable type is fixed to 8-bit integer (INT8). Further, the maximum number of MAC operations for each inference is limited to 30 MMAC (million multiply-accumulate operations), while the length of the audio for inference is fixed to 1 second. In addition to these system complexity constraints, the proposed ASC system is expected to be robust across multiple recording devices and locations similar to the previous editions.

We address this ASC task by utilizing a multi-task learning (MTL) framework, where the classification of recording devices is used as an additional task to aid the ASC task. Our proposed CNN model architecture is designed such that it satisfies the system complexity requirements in terms of MAC operations and total number of parameters. To further improve generalization ability of the model to unseen devices, we employ multiple data augmentation techniques. The quantization-aware-training (QAT) and TFLite tools are then utilized to convert the proposed CNN model to INT8 representation for performing final inference and comparisons to other competing systems along with the challenge baseline.

In the following sections, we first describe the proposed MTL framework in Section 2. The details of the experimental setup and results with analysis are reported in Section 3 and Section 4, respectively. Finally, the conclusions are presented in Section 5.

2. DEVICE CLASSIFICATION-AIDED ASC

A majority of the systems developed previously to address ASC using multiple devices have employed an STL framework. In comparison, the use of MTL frameworks for ASC task is less explored. Generally, an MTL framework consists of an encoder block with shared layers, which are subsequently connected to multiple branches with task-specific layers. A solution based on MTL framework was recently proposed for the ASC task of DCASE. The authors of [13] used an additional task of classifying the audio clip to 3 broader acoustic scenes, in addition to the given task of classifying the audio clip to 10 acoustic scenes for joint training. The 3 broader

[1] https://dcase.community/challenge2022/

acoustic scenes are higher-level abstractions of the 10 scenes, which make these tasks highly related to each other.

In DCASE 2022 Task 1 on low-complexity ASC, the development dataset consists of audio samples from 3 real-devices (A, B, C) and 6 simulated devices (S1-S6). In our proposed MTL framework, we leverage recording device information by classifying recording devices corresponding to the given audio clip as a task, in addition to the ASC task. Specifically, we classify the given audio clip to be recorded using one among the seen real-devices (A, B, C) and simulated/unseen devices. All the simulated devices (S1-S6) and unseen devices are grouped to 1 device-class, thereby forming a total of 4 device-classes. By enabling the shared layers in the encoder block to learn complementary features to classify the recording device corresponding to a given audio clip, we envisage that these layers of the network can learn low-level features which could improve the performance of the model for multi-device ASC task.

One of the previous works, explored the use of device classification for ASC as well [14]. However, for joint training using MTL, the one-hot encoding representation of the predicted device is expanded and converted as a feature map to the multiple CNN layers in the ASC branch. Even though the use of such device-information conditioned MTL training shows improved performance compared to STL framework, such an approach would require the device-classification branch to be included during the inference stage as well. As such, it would increase the number of parameters and MAC operations of the inference model for the given ASC task. To ensure that the feature maps of the shared layers can leverage similar information without increasing the computation complexity, instead of utilizing the predicted device as a feature map, we combine the loss corresponding to the device classification task with the loss of the ASC task for joint training.

We use a combined loss function, which is the weighted loss function from the ASC branch and the device classification branch to perform the joint training of both these tasks. It can be expressed mathematically as

$$\mathcal{L}_{\text{MTL}} = \beta \times \mathcal{L}_{\text{device}} + (1 - \beta) \times \mathcal{L}_{\text{ASC}}, \tag{1}$$

where β is the trade-off parameter that controls the weighted loss. $\mathcal{L}_{\text{device}}$ and $\mathcal{L}_{\text{ASC}}$ corresponds to the 4-device classification loss and the 10-scene classification loss, respectively. Setting $\beta = 0$ is similar to the STL framework.

The use of MTL framework with joint training utilizing the weighted loss in Eq (1) for the given ASC task has the additional advantage that once the MTL-based model is trained, the device classification branch can be removed from the model architecture. During the inference stage, only the 10-scene classification branch is utilized. Therefore, the number of parameters and MAC operations remains the same as that of the STL framework.

2.1. CNN model architecture

Our proposed MTL framework based on CNN architecture is shown in Figure 1. It consists of two branches, i.e., a branch corresponding to the 10-scene classification task and another branch corresponding to the 4-device classification task. The 10-scene classification branch consists of a combination of separable and strided convolutions. We use $L = 5$ convolution layers with number of filters set as $(150, 140, 160, 180, 220)$. It is noted that separable convolutions are used for all the layers except the first convolution layer. In addition, a stride of 2 is used for all convolutional layers except the last convolution layer. Strided convolutions are applied uniformly

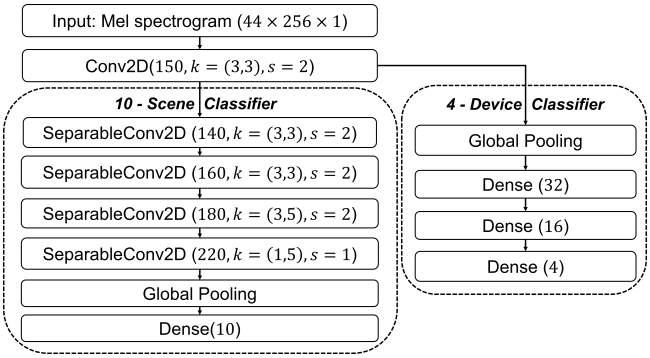

Figure 1: The proposed MTL framework with separate branches for 10-scene classification and 4-device classification task. Conv2D and SeparableConv2D $(n, (p \times q), s)$ represents 2D convolution and separable convolution operation with n filters of kernel size $p \times q$ with a stride of s.

across both frequency and time dimensions. The kernel size for the first three convolution layers is chosen as (3×3), whereas we consider a kernel size of (3×5) and (1×5) for the fourth and fifth convolution layers, respectively. Finally, global pooling operation is performed to gather all the components from the last convolution layer. The output layer consists of a dense layer with 10 units corresponding to the number of acoustic scene classes which undergoes softmax$(\cdot)$ operation to obtain the scene prediction probabilities.

For the 4-device classification branch, the output of the first convolution layer undergoes a global pooling operation and is connected to two dense layers of 32 and 16 units each. The output layer consists of a dense layer with softmax$(\cdot)$ operation with 4 units corresponding to the device classes. Rectified linear unit (ReLU) is chosen as the activation function and a weight decay of $1\text{e}{-5}$ is applied for all convolution layers. We also use dropout of 0.5 after the $L = 2, 4, 5$ convolution layers and a dropout of 0.3 between the dense layers of the 4-device classification branch. The network parameters for both the branches are determined empirically.

2.2. INT8 Quantization

As part of the Task 1 constraints, it is required that the model and the input data used for inference should use INT8 variable type. To minimize the drop in performance after the quantization step, we first perform quantization-aware-training (QAT) utilizing TensorFlow framework with a subset of the training data [15]. After the model is fine-tuned using QAT, it is converted to INT8 TFLite model. The number of parameters and the MAC operations of the TFlite model is computed by using the scripts in Nessi[2] toolkit, which is provided by the Task 1 challenge organizers. The proposed TFLite model has $127,236$ parameters with 27.53 MMAC operations, which satisfies the model size and complexity constraints.

2.3. Data augmentation

To ensure that the proposed model is generalized to recordings using an unseen device or a recording from an unseen location, multiple data augmentation techniques that are well-explored in the past are used [16, 17]. In addition, we also explore the use of AugMix [18]

[2] https://github.com/AlbertoAncilotto/NeSsi

Table 1: Performance comparison of the STL and the proposed MTL framework for various β. The device-wise log loss is shown in columns A-S6. The best results are shown in bold.

Framework	Average Log Loss	Average Accuracy (%)	A	B	C	S1	S2	S3	S4	S5	S6
STL	1.333 ± 0.012	51.25 ± 0.33	0.98	1.24	**1.11**	1.37	1.35	1.35	1.50	1.45	1.61
MTL, $\beta = 0.1$	1.327 ± 0.009	51.65 ± 0.31	0.99	1.24	**1.11**	1.36	1.35	**1.33**	1.48	1.45	1.61
MTL, $\beta = 0.2$	$\mathbf{1.319 \pm 0.017}$	$\mathbf{51.98 \pm 0.55}$	**0.97**	**1.22**	**1.11**	1.36	**1.34**	1.34	1.48	**1.43**	**1.58**
MTL, $\beta = 0.3$	1.323 ± 0.014	51.31 ± 0.55	**0.97**	**1.22**	**1.11**	**1.35**	**1.34**	1.34	**1.47**	1.46	1.61
MTL, $\beta = 0.4$	1.347 ± 0.014	50.53 ± 0.54	0.99	1.25	1.13	1.39	1.37	1.37	1.49	1.47	1.63
MTL, $\beta = 0.5$	1.335 ± 0.010	51.31 ± 0.40	0.98	1.25	1.12	1.38	1.37	1.36	**1.47**	1.45	1.59

and RawBoost [19] technique that have not yet been extensively used for ASC task. Since device classification is utilized in the MTL framework, it is important to assign the device labels according to the augmentation technique being employed. The augmentation techniques utilized are summarised below.

- **Time stretch (TS) :** The audio sample is either slowed down or sped up without altering the pitch by a factor randomly chosen from the uniform distribution $[0.8, 1.2]$.
- **Time shift (TSH) :** A random start time is chosen from the 1 second duration of the given audio clip to obtain two audio segments. The augmented audio clip is the appended audio segments shifted in time.
- **Block mixing (BM) :** For a given audio clip corresponding to a particular device and acoustic scene, an additional audio clip recorded with the same device and belonging to the same acoustic scene is selected randomly. The augmented audio clip is the weighted average of the given audio clip and the additional audio clip.
- **AugMix :** Similar to the AugMix strategy used for image data augmentation, a given audio clip undergoes multiple chains of transformations using the above data augmentation steps (time stretch, time shift, block mix). At the final stage, the original audio clip is combined with the transform-chain augmented audio clip to obtain the augmented audio sample. We set the default values of maximum number and depth of the transform chain to 3 with the α parameter of both Dirichlet and Beta distribution set to 1.0 as in [18].
- **RawBoost :** The recently proposed data augmentation technique RawBoost for anti-spoofing is considered for exploration in the given ASC task. In RawBoost, multiple notch filters with randomly chosen center frequencies and bandwidths are used to design an FIR filter. The use of FIR filter on the given audio clip, introduces variations on the audio spectrum, which could improve generalization of the model for unseen device. We set the similar filter design configurations as used in [19].

Among the above data augmentation techniques, time stretch, time shift, block mixing and AugMix does not change the frequency characteristics of the given audio clip. Therefore, they are device-label invariant techniques. However, the RawBoost technique applies FIR filtering on the given audio clip which modifies its frequency characteristics. As such, it is a device-label variant technique and these clips are labeled as belonging to the simulated and unseen device class. Next, we discuss the experimental setup for the studies conducted in this work.

3. EXPERIMENTAL SETUP

We use the TAU Urban Acoustic Scenes 2022 Mobile, development dataset provided as part of the challenge for the studies in this work [20]. This development set consists of an official training and validation split with audio clips of 1 second in duration sampled at 44.1 kHz. The clips correspond to audio scene recordings from 10 cities using 9 devices: 3 real devices (A, B, C) and 6 simulated devices (S1-S6). The simulated device recordings (S1-S6) are obtained by applying impulse responses of real devices and additional dynamic range compression on recordings from device A. There are $102,150$ clips from device A, while for devices (B, C, S1-S3) there are $\approx 7,500$ clips each. It should be noted that the audio clips corresponding to devices S4-S6 does not appear in the training split and is only present in the validation split. The 10 audio scenes to be classified 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"}.

We do not utilize any external data or pre-trained models for training our proposed system. For each audio clip, we compute the short-time Fourier transform (STFT) of the signal by using a frame length of 2048 samples and hop length of 1024 samples. After computing the STFT, the corresponding log Mel-spectrogram is computed with 256 Mel-bins. We then obtain a time-frequency (TF) representation of each audio clip with dimension 44×256, which is provided as the input feature to the proposed MTL framework.

During training, the optimization is performed using the Adam optimizer [21], with an initial learning rate of 0.001 and a maximum epoch of 200 with a batch size of 32 samples. The learning rate is reduced by a factor of 0.1 if the validation loss does not decrease after 5 epochs. Early stopping method is used to stop the training if the validation loss does not decrease after 10 epochs. The categorical focal loss [22] is chosen as the loss function for both classification tasks. For all the evaluations, we utilize the validation split from the development set. The multi-class cross-entropy (log loss) is used as the primary metric for evaluating system performance along with average of the class-wise accuracy as a secondary measure. The performances are evaluated over 5 trials and the average results are reported.

4. RESULTS AND ANALYSIS

This section discusses the results and their analysis conducted for various studies in this work. The impact of proposed MTL framework and data augmentation on ASC and comparison to other systems are reported in the following subsections.

Table 2: Performance comparison for various data augmentation approaches and their combination. The best results are shown in bold.

Method	Avg. Log Loss	Avg. Acc. (%)	A	B	C	S1	S2	S3	S4	S5	S6
No Data Augmentation	1.319 ± 0.017	51.98 ± 0.55	0.97	1.22	1.11	1.36	1.34	1.34	1.48	1.43	1.58
TS + TSH + BM (1)	1.314 ± 0.007	52.08 ± 0.33	0.96	1.21	1.13	1.36	1.34	1.32	1.46	1.44	1.56
AugMix (2)	1.309 ± 0.014	52.45 ± 0.56	0.96	1.23	1.12	1.35	1.33	1.33	1.43	1.43	1.56
RawBoost (3)	1.298 ± 0.010	52.22 ± 0.71	0.99	1.18	1.10	1.33	1.33	1.31	1.41	1.41	1.58
Combined (1) + (2) + (3)	$\mathbf{1.265 \pm 0.009}$	$\mathbf{53.59 \pm 0.43}$	**0.94**	**1.17**	**1.07**	**1.29**	**1.32**	**1.28**	**1.39**	**1.40**	**1.50**

4.1. Performance evaluation of STL and MTL framework

We begin our experiments by evaluating the performance of the ASC system using the STL and MTL framework. For both these frameworks, the samples from the training split of the development dataset are used without applying any specific data augmentation technique. For MTL framework, the trade-off parameter, β is varied from 0.1 to 0.5. Since our primary task is the 10-scene classification, we set the maximum value of β as 0.5, which corresponds to equal weight to the loss of both tasks. The average log loss and the average accuracy obtained using the STL and MTL framework for various β are shown in Table 1. It can be seen that the MTL framework with $\beta = 0.2$ achieves the best performance with an average log loss of 1.319 and an average accuracy of 51.98%. In comparison, the STL framework achieves an average log loss of 1.333 and an average accuracy of 51.25%. From the results belonging to device-wise log loss, it can be seen that the use of MTL framework achieves comparatively lower log loss for most of the seen and unseen devices. In addition, we note that the MTL framework with $\beta = 0.2$ achieves an average accuracy of $\approx 97\%$ on the 4-device classification task. This indicates that enabling the shared layer to learn device-specific low-level features helps to improve the ASC task. Since the MTL framework with $\beta = 0.2$ achieves the best ASC performance, we use the same for the rest of our experiments.

4.2. Performance evaluation of data augmentation techniques

In this set of experiments, we evaluate the performance of the proposed MTL framework using various data augmentation methods presented in Section 2.3. A given data augmentation technique is applied separately on all the samples in the training dataset for comparing its effect on the ASC task. We apply the time stretch, time shift and block mixing with equal probability and combine them as one data augmentation technique (TS + TSH + BM). The average log loss, average accuracy as well as the device-wise log loss obtained using none of the data augmentation techniques and applying each of them separately is shown in Table 3. It can be seen that each augmentation technique helps to improve the average log loss and accuracy. Since the objective of RawBoost technique is to generate unseen/simulated devices, its performance especially on devices S4 and S5 is improved, while performance on device A is degraded. Subsequently, all the data augmentation techniques are combined to train our final model. It achieves an average log loss and average accuracy of 1.265 and 53.59% respectively, which are better than those obtained with any of the individual data augmentation methods as well as those without any augmentation.

4.3. Comparison to other systems

We now compare the performance of the proposed MTL-based ASC system on the development set with the well performing systems submitted to the DCASE Task 1 challenge on low-complexity ASC as well as the challenge baseline. To this extent, we perform INT8 quantization on our best performing model as described in Section 2.2 for inference. The INT8 quantized model achieves an average log loss 1.273 and an average accuracy of 53.5%. The reported performances of other systems and the baseline, sorted with reference to average log loss are shown in Table 3. It can be seen that we achieve a comparable performance with other systems with a significant improvement over the baseline system. We note that most of the proposed systems are based on STL frameworks, which utilize knowledge distillation technique for model design. However, Sugahara *et al.* [23] uses an MTL framework with the 3 broader ASC as the additional task. It is also noted that the baseline system uses a 3-layer CNN using log Mel-spectrograms with $46,512$ parameters and 29.23 MMACS [12].

Table 3: Performance comparison of the proposed MTL-based system with other well performing systems on the development set of DCASE 2022 Task 1.

System	Avg. Log Loss	Avg. Accuracy (%)
Lee *et al.* [24]	0.835	70.1
Anastácio *et al.* [25]	1.103	60.5
Schmid *et al.* [26]	1.139	58.0
Sugahara *et al.* [23]	1.182	56.5
Kim *et al.* [27]	1.259	54.0
Proposed MTL **(Ours)**	1.273	53.5
Morocutti *et al.* [28]	1.288	52.7
Xin *et al.* [29]	1.295	60.3
Yu *et al.* [30]	1.305	51.7
Shao *et al.* [31]	1.360	54.1
Baseline [12]	1.575	42.9

5. CONCLUSIONS

In this paper, we explore the use of an MTL framework with the device classification task as an additional task to address the DCASE 2022 Task 1 on low-complexity ASC. The proposed CNN model is designed such that it meets the Task 1 model complexity constraints. We also explore the use of various data augmentation techniques to improve the generalization of the model. Evaluations on the development set show that the use of MTL framework along with the various data augmentation techniques help to improve the performance of the ASC task.

6. REFERENCES

[1] D. Wang and G. J. Brown, *Computational auditory scene analysis: Principles, algorithms, and applications*. Wiley-IEEE press, 2006.

[2] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational analysis of sound scenes and events*. Springer, 2018.

[3] L. Sicong, Z. Zimu, D. Junzhao, S. Longfei, J. Han, and X. Wang, "UbiEar: Bringing location-independent sound awareness to the hard-of-hearing people with smartphones," *in Proc. ACM Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, vol. 1, no. 2, pp. 1–21, 2017.

[4] R. Radhakrishnan, A. Divakaran, and A. Smaragdis, "Audio analysis for surveillance applications," in *Proc. IEEE Workshop Applicat. Signal Process. Audio Acoust.* IEEE, 2005, pp. 158–161.

[5] S. Suh, S. Park, Y. Jeong, and T. Lee, "Designing acoustic scene classification models with CNN variants," DCASE2020 Challenge, Tech. Rep., June 2020.

[6] B. Kim, S. Yang, J. Kim, and S. Chang, "Domain generalization on efficient acoustic scene classification using residual normalization," in *Proc. DCASE Workshop*, 2021, pp. 21–25.

[7] M. Kośmider, "Spectrum correction: Acoustic scene classification with mismatched recording devices," *in Proc. Interspeech*, pp. 4641–4645, 2020.

[8] C.-H. H. Yang, H. Hu, S. M. Siniscalchi, Q. Wang, W. Yuyang, X. Xia, Y. Zhao, Y. Wu, Y. Wang, J. Du, and C.-H. Lee, "A lottery ticket hypothesis framework for low-complexity device-robust neural acoustic scene classification," DCASE2021 Challenge, Tech. Rep., June 2021.

[9] H. Hee-Soo, J. Jee-weon, S. Hye-jin, and L. Bong-Jin, "Clova submission for the DCASE 2021 challenge: Acoustic scene classification using light architectures and device augmentation," DCASE2021 Challenge, Tech. Rep., June 2021.

[10] K. Koutini, S. Jan, and G. Widmer, "CPJKU submission to DCASE21: Cross-device audio scene classification with wide sparse frequency-damped CNNs," DCASE2021 Challenge, Tech. Rep., June 2021.

[11] N. Zhao, "Low-complexity acoustic scene classification using knowledge distillation and multiple classifiers," DCASE2021 Challenge, Tech. Rep., June 2021.

[12] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in DCASE 2022 challenge," *arXiv preprint arXiv:2206.03835*, 2022.

[13] H.-j. Shim, J.-w. Jung, J.-h. Kim, and H.-J. Yu, "Attentive max feature map and joint training for acoustic scene classification," in *Proc. ICASSP*. IEEE, 2022, pp. 1036–1040.

[14] Z. Ren, Q. Kong, J. Han, M. D. Plumbley, and B. W. Schuller, "CAA-Net: Conditional atrous CNNs with attention for explainable device-robust acoustic scene classification," *IEEE Transactions on Multimedia*, vol. 23, pp. 4131–4142, 2020.

[15] R. Krishnamoorthi, "Quantizing deep convolutional networks for efficient inference: A whitepaper," *arXiv preprint arXiv:1806.08342*, 2018.

[16] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal processing letters*, vol. 24, no. 3, pp. 279–283, 2017.

[17] H. Hu, C.-H. H. Yang, X. Xia, X. Bai, X. Tang, Y. Wang, S. Niu, L. Chai, J. Li, H. Zhu, *et al.*, "A two-stage approach to device-robust acoustic scene classification," in *Proc. ICASSP*. IEEE, 2021, pp. 845–849.

[18] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "AugMix: A simple method to improve robustness and uncertainty under data shift," in *Proc. ICLR*, 2020.

[19] H. Tak, M. Kamble, J. Patino, M. Todisco, and N. Evans, "RawBoost: A raw data boosting and augmentation method applied to automatic speaker verification anti-spoofing," in *Proc. ICASSP*. IEEE, 2022, pp. 6382–6386.

[20] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in DCASE 2020 challenge: generalization across devices and low complexity solutions," in *Proc. DCASE Workshop*, 2020, pp. 56–60.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[22] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. ICCV*, 2017, pp. 2980–2988.

[23] R. Sugahara, R. Sato, M. Osawa, Y. Yuno, and C. Haruta, "Self-ensemble with multi-task learning for low-complexity acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[24] J.-H. Lee, J.-H. Choi, P. M. Byun, and J.-H. Chang, "HYU submission for the DCASE 2022: Efficient fine-tuning method using device-aware data-random-drop for device-imbalanced acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[25] R. Anastácio, L. Ferreira, F. Mónica, and C. B. Luís, "Ai4edgept submission to DCASE 2022 low complexity acoustic scene classification task1," DCASE2022 Challenge, Tech. Rep., June 2022.

[26] F. Schmid, S. Masoudian, K. Koutini, and G. Widmer, "CPJKU submission to DCASE22: Distilling knowledge for low-complexity convolutional neural networks from a patchout audio transformer," DCASE2022 Challenge, Tech. Rep., June 2022.

[27] T. Kim, G. Lee, and J. Park, "KT submission for the DCASE 2022 challenge: Modernized convolutional neural networks for acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[28] T. Morocutti and D. Shalaby, "Receptive field regularized CNNs with traditional audio augmentations," DCASE2022 Challenge, Tech. Rep., June 2022.

[29] Y. Xin, Y. Zou, F. Cui, and Y. Wang, "Low-complexity acoustic scene classification with mismatch-devices using separable convolutions and coordinate attention," DCASE2022 Challenge, Tech. Rep., June 2022.

[30] J. Yu, R. Shi, T. He, and K. Guo, "Acoustic scene classification based on feature fusion and dilated-convolution," DCASE2022 Challenge, Tech. Rep., June 2022.

[31] Y.-F. Shao, X. Zhang, G.-G. Bing, K.-M. Zhao, J.-J. Xu, Y. Ma, and W.-Q. Zhang, "Mini-segnet for low-complexity acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.