HYU SUBMISSION FOR THE DCASE 2022: FINE-TUNING METHOD USING DEVICE-AWARE DATA-RANDOM-DROP FOR DEVICE-IMBALANCED ACOUSTIC SCENE CLASSIFICATION

Technical Report

Joo-Hyun Lee, Jeong-Hwan Choi*, Pil Moo Byun*, and Joon-Hyuk Chang*

Department of Electronic Engineering, Hanyang University, Seoul, Republic of Korea
{jhyun42, brent1104, fordream0309, jchang}@hanyang.ac.kr

ABSTRACT

This paper address the Hanyang University team submission for the DCASE 2022 Challenge Low-Complexity Acoustic Scene Classification task. The task aims to design a generalized audio scene classification system for various devices under low complexity and short input time conditions. We follow two strategies to achieve our goal: improving the model structure for short segmented audio and adopting transfer learning methods that are generalizable to unknown devices. Based on the BC-ResNet, which showed the best performance in DCASE 2021 challenge, we incorporate the method proposed in the field of short-duration speaker verification to secure high accuracy. In addition, we propose a novel fine-tuning method using device-aware data-random-drop to get a generalized model across multiple devices. Most of the training dataset is data recorded with one specific device. Therefore, we devised a fine-tuning method that gradually excludes data recorded with a specific device from mini-batch during training. Following the official protocol of cross-validation setup from the TAU Urban Acoustic Scenes 2022 Mobile development dataset, we achieved 70.1% accuracy and 0.835 multi-class cross-entropy loss, respectively.

Index Terms— Acoustic scene classification, device imbalance, fine-tuning, short audio, knowledge distillation, quantization aware training

1. INTRODUCTION

Among the various intriguing tasks DCASE2022 challenges, we focus on Task 1: Low-Complexity Acoustic Scene Classification with Multiple Devices [1,2]. Audio Scene classification (ASC) is the task that has the objective of categorizing sound scenes. ASC is applied in various fields, including context-aware services, surveillance, and improving the performance of audio event detection tasks [3–5]. This year is similar to DCASE 2021 Challenge Task 1A in many aspects. However, unlike last year, the duration of the audio data is shortened to 1 second, so the acoustic scene is inferred only from a short segment [6, 7]. Additionally, a limit on the number of allowed multiply-accumulate operations (MACs) has been added; the number of MACs should be up to 30M. The requirements for system complexity are also different from 2021 Task 1. The number of model parameters needs to be smaller than 128K, and the model parameters variable type should be fixed into INT8.

In this work, we propose a lightweight model that effectively works on short segmented audio and a learning strategy that can increase generalization capabilities for multiple devices. First, we introduce the BC-Res2Net backbone structure converted from the Res2Net structure [8] into broadcast learning [9] to train multi-scale representations. Moreover, multi-scale frequency channel attention (MFA) structure [10] and feature pyramid module (FPM) [11], proposed in the speaker verification field and are effective for extracting and aggregating the features from short speech signals, are applied to the BC-Res2Net based ASC model. Second, we suggest a novel fine-tuning method using device-aware data-random drop, which improves the generalization ability across multiple devices by excluding several data of the selected device in batch-level processing. We selected a specific device with the most records from the training dataset. Then, we increase the generalization capability for unseen devices by gradually excluding the data recorded with specific selected devices from the batch. We also applied quantization-aware training [12] and knowledge distillation [13] to reduce memory use and misclassification in confusing pairs of scenes, respectively The proposed methods were evaluated on the official cross-validation setup of the TAU Urban Acoustic Scenes 2022 Mobile development dataset.

The remainder of this paper is organized as follows. Sections 2 and 3 describe the proposed ASC model architecture and efficient fine-tuning strategies. Section 4 presents the experimental setup. Finally, the results and discussion of the study are in Section 5.

2. ASC SYSTEM FRAMEWORK

2.1. Datasets

DCASE 2022 Challenge Task 1 allows using the TAU Urban Acoustic Scenes 2022 Mobile development dataset [1]. All audios are in a single-channel 44.1kHz 24-bit format, and each segment is cropped into 1 second. Audios are recorded with three real devices (A, B, and C) and six simulated devices (S1-S6). The development dataset consists of 230,350 audio segment recordings from 10 cities in 10 acoustic scenes: ”airport,” ”shopping mall,” ”metro station,” ”street pedestrian,” ”public square,” ”street with traffic,” ”park,” and traveling by ”tram,” ”bus,” and ”metro.” The dataset is separated with a provided validation policy [2] into training and test subsets consisting of 139,970 and 29,680 segments, respectively. In the training subset, data of device A occupies 73% of the total, whereas in the test set, the data from all devices are evenly distributed, and devices S4-S6 are unseen in training. The evaluation set uses an additional two cities and five devices. Specifically, real device D and simulated devices (S7-S11) are not included in the development data.

*Equal contributions.

2.2. Preprocessing

We extract input features through three processes: down-sampling to 16kHz, acoustic feature extraction, and data augmentation. Specifically, the acoustic features are 256-dimensional log Mel spectrograms extracted with 2048 samples of Hanning window and 512 samples shift. Input features are randomly rolled along the temporal axis with a range of -0.5 to 0.5 seconds and pasted the out-of-range part to the opposite side. Specaugment [14] with two frequency and two temporal masks are adopted for data augmentation, and each is applied with a probability of 0.8. Mask parameters of 40 and 4 are used for frequency masks and temporal masks, respectively. Mixup [15] with α = 0.3 is also applied to the acoustic feature space.

2.3. Proposed ASC model architecture

Broadcast residual learning [9] extracts two feature maps specific to frequency and temporal dimension through frequency-wise 2D and temporal-wise 1D convolution. These feature maps are combined with backbone structures such as ResNet [16] and Transformer [17] to be used as state-of-the-art models in keyword spotting, speaker recognition, and ASC fields [9, 18, 19]. In particular, in DCASE 2021 challenge Task 1A, the system with the best performance modified the BC-ResNet [9] by using max-pool layers and limiting the receptive field. Also, ResNorm was proposed to perform normalization for each frequency band in the residual path [19]. In this work, we adopt the concept of splitting the bottleneck convolution block of the Res2Net structure to the BC-ResNet since the Res2Net achieves more efficient computations than the traditional convolutions by constructing hierarchical residual-like connections within a single residual. To be specific, we divided the feature maps into four equal subsets. Each feature subset has the same frequency and temporal dimension as the input feature map but 1/4 the number of channels. Except for the initial subset, each has a corresponding convolution. The feature subset is added with the previous output of the subset convolution and then fed into the subsequent convolution. Note that all frequency-wise 2D convolution and temporal-wise 1D convolution of BC-ResNet block to Res2Net style. Therefore, BC-Res2Net not only effectively obtains frequency and temporal features by Broadcast learning but also operates in a multi-scale manner with low computational complexity.

Recently, in speaker verification, the method of reweighting the input feature map using the attention mechanism and aggregation method, which merged the features from multiple stages of the backbone structure into one, were studied to overcome the degrade under short utterance scenarios. MFA and FPM are representative methods, and we modify the networks to be appropriate for the audio scene classification task and then combine them with BC-Res2Net. The overall proposed architecture is shown in Table 1. In [10], MFA was applied only before feeding into the backbone network. We also reweight the output using ResNorm followed by MFA for every four stages, composing BC-Res2Net. Reweighted outputs via MFA are upsampled with the same size as the output of stage 1. Then aggregated into a single feature map in a bottom-up pathway by FPM [11].

Table 1: Architectures of proposed BC-Res2Net based ASC models. C, F, and T denote the number of convolution channel, frequency bins, and time sequences, respectively. Input feature size is $1 \times F \times T$

Output size	Stage	Operator
$2C \times F/2 \times T/2$	Stem	Conv2D $\lfloor 5 \times 5 \rfloor$, stride 2 BatchNorm + MFA
$C \times F/2 \times T/2$	Stage 1	BC-Res2Net $\times$ 2 ResNorm + MFA
$1.5C \times F/4 \times T/4$	Stage 2	Max-pool $\lfloor 2 \times 2 \rfloor$ BC-Res2Net $\times$ 2 ResNorm + MFA
$2C \times F/8 \times T/8$	Stage 3	Max-pool $\lfloor 2 \times 2 \rfloor$ BC-Res2Net $\times$ 2 ResNorm + MFA
$2.5C \times F/8 \times T/8$	Stage 4	BC-Res2Net $\times$ 3 ResNorm + MFA
$4C \times 1 \times 1$	Aggregation	FPM
$10 \times 1 \times 1$	Classifier	Linear

3. TRAINING STRATEGY

3.1. Quantization Aware Training

We have to apply quantization to fix the parameter variable type into INT8. We utilized quantization aware training (QAT) [12, 20], which performs floating-point calculations during training but simulates the effect of INT8 with a fake quantization module through clamping and rounding. We used the QAT plugin provided by the PyTorch-lightning library [21] for QAT. Default values were used for all QAT settings, and layer fusion was applied to all Convolution-BatchNorm-ReLU sequences in the model. After QAT was completed, the inference was performed with the quantized converted model.

3.2. Knowledge Distillation

Knowledge distillation (KD) [13, 22, 23] have been shown effective for the ASC task to maintain the performance even with low complexity. The KD is generally used for model compression, but we used it for generalization to devices, as in [19]. We trained the teacher model using the entire development set without validation, and the student model was trained on the original cross-validation setup. The experiment was conducted as a case where the size of the teacher model was set to twice the size of the student model and a case where the channel size was set to be the same as the student model.

3.3. Proposed fine-tuning method using device-aware data-random-drop

The ASC model spends more effort representing device A than other devices since device A takes up a big percentage of the training dataset. This can reduce the generalization power across different devices. In this work, we use a fine-tuning strategy with device-aware data-random-drop to mitigate the data imbalance problem. First, regular model training is performed until the validation loss converges and model parameters are saved. During fine-tuning, saved parameters are loaded except for the classifier layer. Then, fine-tuning begins with the device-aware data-

Table 2: Overall top-1 test accuracy and device-wise log loss comparison of KD and proposed fine-tuning method on the TAU Urban Acoustic Scenes 2022 Mobile development dataset. (Acc. indicates top-1 test accuracy)

Systems	**KD**	**Fine-tuning** (λ)	**Device**									**Average** log loss / Acc.
			A	B	C	S1	S2	S3	S4	S5	S6	
BC-Res2Net w/ MFA w/ FPM	✗	✗	0.938	1.065	1.007	1.203	1.217	1.122	1.610	1.350	1.395	1.193 / 60.3%
	✗	0.04	0.769	0.969	0.924	1.082	1.164	0.997	1.423	1.191	1.273	**1.072 / 62.2%**
	✗	0.4	0.797	0.963	0.906	1.050	1.185	0.972	1.371	1.185	1.290	**1.066 / 62.6%**
	large (C=80)	✗	0.955	1.059	1.029	1.164	1.288	1.048	1.452	1.225	1.358	1.157 / 61.7%
		0.4	0.709	0.950	0.875	0.970	1.103	0.924	1.287	1.112	1.253	**1.005 / 64.9%**
	small (C=40)	✗	0.617	0.771	0.762	0.853	0.854	0.794	1.240	0.910	1.130	0.873 / 69.2%
		0.4	0.585	0.759	0.709	0.833	0.855	0.746	1.240	0.899	1.070	**0.835 / 70.1%**

random-drop method. The proposed method randomly drops the labeled data with device A in every mini-batch. As a result, the model can achieve generalization for multiple devices by using other devices(B-S3) more often than pre-trained. However, fine-tuning with few data of device A lead to ASC performance degradation since the number of training data is significantly reduced. Therefore, we design the method to become gradually more fit with minority devices by increasing the drop rate step by step. We configure the drop rate to increase along with the shape of the sigmoid function at every epoch from 0 to the given parameter. Furthermore, we add regularization to the loss to prevent overfitting for other devices(B-S3). The regularization focuses on minimizing the square weight difference between the pre-trained model and the model after fine-tuning to avoid excessive device A information loss and class collapse.

4. EXPERIMENTAL SETUP

We set the channel size C of the proposed BC-Res2Net ASC model as 40. The subspectral normalization [24] with four numbers of subbands and ResNorm with 0.1, the hyperparameter of the identity shortcut path, were also applied to BC-Res2Net. In the pre-training phase, teacher and student models were trained for 300 epochs using the AdamW optimizer [25] with a weight decay of 0.05, and the mini-batch size was set to 512. The warmup [26] was applied where the learning rate linearly increased from 1e-8 to 0.01 over the first ten epochs and decayed to zero with a cosine annealing scheduler [27]. However, in the fine-tuning phase, an AdamW optimizer with weight decay of 1e-8 and a fixed learning rate of 1e-5 was used. The device A data were excluded with a maximum drop rate of 0.9 from every mini-batch by applying the device-aware data random drop method. The scaling factor of regularization λ was set to 0.04 and 0.4, and the results for each case were submitted. The mixup was disabled to correct the mismatch between training and test conditions. The quantization-aware training was applied in both training phases, and all experiments were implemented using the PyTorch library [28] and trained using a single NVIDIA RTX 3090 GPU with 24GB memory. We evaluated the result in terms of top-1 test accuracy and multi-class cross-entropy (log loss) since the log loss had been chosen as the evaluation metric of the competition.

5. RESULT & DISCUSSION

The DCASE 2022 challenge allowed submitting four results. We submitted four BC-Res2Net-based models trained with different training conditions are as follows:

- The model without KD, fine-tuning with λ of 0.04.
- The model without KD, fine-tuning with λ of 0.4.
- The model with large teacher model KD, fine-tuning with λ of 0.4.
- The model with small teacher model KD, fine-tuning with λ of 0.4.

The proposed BC-Res2Net with MFA and FPM had 126.6K model parameters and 27.76 million MACs, and it satisfied the competition conditions. Table 2 summarizes the result of four systems based on the proposed model architecture that we submitted. We also compared the performance of the submitted model and the model without fine-tuning. Note that teacher models were trained with both train and test datasets. The teacher models generated a more fitting feature on the test dataset, and student models were trained to track it; Thus, the overall device performance increased. Comparing the models with KD, the using smaller size teacher obtained better performance than the larger size teacher.

The models which were trained with the proposed fine-tuning method outperformed the pre-trained model in both accuracy and log loss both with and without KD modeling. For the models with fine-tuning only, λ of 0.04 achieved relative improvements of 10.6% and 3.2% in terms of average log loss and accuracy compared with the pre-trained model. Especially device-wise log loss on unseen (S4-S6) relative improved by 11.8%, 11.8%, and 8.7%, respectively. These results support that our proposed fine-tuning method benefited the generalization of multiple devices. Similar results were also obtained where KD with a large or small teacher model was applied.

6. REFERENCES

[1] 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," in *arXiv:2206.03835*, 2022.

[2] http://dcase.community/challenge2022/.

[3] M. Valenti, A. Diment, G. Parascandolo, S. Squartini, and T. Virtanen, "Dcase 2016 acoustic scene classification using convolutional neural networks," in *Proc. Workshop Detection Classif. Acoust. Scenes Events*, 2016, p. 95–99.

[4] R. Radhakrishnan, A. Divakaran, and A. Smaragdis, "Audio analysis for surveillance applications," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2005.*, 2005, pp. 158–161.

[5] S. Chu, S. Narayanan, and C.-C. J. Kuo, "Environmental sound recognition with time–frequency audio features," *IEEE*

Transactions on Audio, Speech, and Language Processing, vol. 17, no. 6, pp. 1142–1158, 2009.

[6] I. Martín-Morató, A. Ancilotto, T. Heittola, A. Mesaros, and T. Virtanen, "Low-complexity acoustic scene classification for multi-device audio: analysis of dcase 2021 challenge systems," in *arXiv:2105.13734*, 2021.

[7] http://dcase.community/challenge2021/.

[8] S.-H. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. Torr, "Res2net: A new multi-scale backbone architecture," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 6, p. 652–662, Feb. 2021.

[9] B. Kim, S. Yang, J. Lee, and D. Sung, "Broadcasted residual learning for efficient keyword spotting," in *Proc. INTERSPEECH*, 2021, p. 4538–4542.

[10] T. Liu, R. K. Das, K. Aik Lee, and H. Li, "Mfa: Tdnn with multi-scale frequency-channel attention for text-independent speaker verification with short utterances," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2022, pp. 7517–7521.

[11] Y. Jung, S. M. Kye, Y. Choi, M. Jung, and H. Kim, "Improving multi-scale aggregation using feature pyramid module for robust speaker verification of variable-duration utterances," in *Proc. INTERSPEECH*, 2020, pp. 1501–1505.

[12] https://pytorch-lightning.readthedocs.io/en/stable/_modules/pytorch_lightning/callbacks/quantization.html.

[13] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[14] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," in *Proc. INTERSPEECH*, 2019, pp. 2613–2618.

[15] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. International Conference on Learning Representations (ICLR)*, 2018.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *arXiv:1512.03385*, 2015.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2017, p. 3830–3834.

[18] J.-H. Choi, J.-Y. Yang, Y.-R. Jeoung, and J.-H. Chang, "Improved cnn-transformer using broadcasted residual learning for text-independent speaker verification," in *Proc. INTERSPEECH*, 2022.

[19] B. Kim, S. Yang, J. Ki, and S. Chang, "Domain generalization on efficient acoustic scene classification using residual normalization," in *Proc. Workshop Detection Classif. Acoust. Scenes Events*, 2021, p. 21–25.

[20] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, p. 2704–2713.

[21] W. Falcon et al., "Pytorch lightning," *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, vol. 3, 2019.

[22] J.-W. Jung, H.-S. Heo, H.-J. Shim, and H.-J. Yu, "Knowledge distillation in acoustic scene classification," *IEEE Access*, vol. 8, pp. 166 870–166 879, 2020.

[23] J. Kim, M. Hyun, I. Chung, and N. Kwak, "Feature fusion for online mutual knowledge distillation," in *Proc. IEEE International Conference on Pattern Recognition (ICPR)*, 2021, p. 4619–4625.

[24] S. Chang, H. Park, J. Cho, H. Park, S. Yun, and K. Hwang, "Subspectral normalization for neural audio data processing," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 850–854.

[25] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. International Conference on Learning Representations (ICLR)*, 2019.

[26] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch sgd: Training imagenet in 1 hour," in *arxiv:1706.02677*, 2017.

[27] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Sgdr: Stochastic gradient descent with warm restarts," in *Proc. International Conference on Learning Representations (ICLR)*, 2017.

[28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2019, pp. 8024–8035.