AUDIO TAGGING AND DEEP ARCHITECTURES FOR ACOUSTIC SCENE CLASSIFICATION: UOS SUBMISSION FOR THE DCASE 2020 CHALLENGE

Technical Report

Hye-jin Shim[*], *Ju-ho Kim*[*], *Jee-weon Jung*, *Ha-jin Yu*[†],

University of Seoul, School of Computer Science, Seoul, South Korea

ABSTRACT

In this technical report, we address the UOS submission for the Detection and Classification of Acoustic Scenes and Events 2020 Challenge Task 1-a. We propose to utilize the representation vectors, extracted from a pre-trained audio tagging system, for the acoustic scene classification task. Audio tagging denotes the existence of various sound events and is known to help the classification of acoustic scene. To select suitable feature for the acoustic scene classification task, we also explore deep architectures such as light convolutional neural networks and convolutional block attention module. Experiments are conducted using the official fold-1 configuration test set. Results using audio tagging representation and deep architectures demonstrate accuracies of 68.8% and 70.5%, compared to that of 65.3% of the baseline. Additionally, score-sum ensemble of the two proposed systems has an accuracy of 71.9% which shows 10.1% relative improvement.

***Index Terms*—** Acoustic scene classification, audio tagging, deep architecture, deep neural network

1. SYSTEM DESCRIPTION

This technical report addresses the submission of the University of Seoul (UOS) team for the Detection and Classification of Acoustic Scenes and Events (DCASE) 2020 Challenge task 1-a [1]. We first introduce our baseline system that inputs Mel-spectrograms and conduct the acoustic scene classification (ASC) task in an end-to-end fashion. Two proposed systems are introduced, where one explore audio tagging representation [2] and deep architectures [3, 4]. Our priority in this report is to detail our choices made on various hyper-parameters for input feature selection, deep neural network (DNN) architecture, and others for making our work reproducible. Hypotheses, inspirations, and other details will be dealt in our academical paper which will be submitted to the DCASE 2020 workshop.

2. INPUT FEATURE EXTRACTION

Across all systems, we use Mel-spectrograms as input to the DNN. Pre-emphasis with 0.97 coefficient and utterance-level mean and variance normalization are applied before and after extracting Mel-spectrograms. Mel-spectrograms are extracted using 128 Mel-filterbanks. The number of fast Fourier transform bin is 2048 and the length of the window and shift size are 40ms and 20ms, respectively. We use the original sampling rate of 44.1 kHz without utilization of any re-sampling methods. We use the *torchaudio* package in PyTorch library, which is a Python-based deep learning toolkit.

[*]Equal contribution

[†]Corresponding author

Table 1: The baseline system architecture (l: length of input sequence).

Type	Filter/Stride	Output
Conv_Block_1	3×3 / 1×1	$l \times 128 \times 16$
SE-Res_1	3×3 / 1×1	$l \times 128 \times 16$
SE-Res_2	3×3 / 3×6	$(l / 6) \times 43 \times 32$
SE-Res_3	3×3 / 5×5	$(l / 30) \times 9 \times 64$
AvgPool	*Global*	64
FC_1	-	64
FC_2	-	10

Table 2: Common hyper-parameters for training models.

Hyper-parameter	
epoch	800
batch size	24
optimizer	SGD
data augmentation	mixup ($a = 0.1$) [5]
initial learning rate	0.001
learning rate scheduling	cosine annealing warm restarts

3. BASELINE ARCHITECTURE

The baseline used in this paper is a variant of the SE-ResNet [6] which is an end-to-end ASC system. Table 1 describes the structure of the baseline. Conv_Block is composed of convolution layer (Conv), batch normalization layer (BN) [7], and rectified linear unit (ReLU) layer. SE-Res corresponds to a sequence of layers, Conv-BN-ReLU-Conv-BN-Squeeze-Excitation, with a residual connection [8]. SE-Res_1, SE-Res_2, and SE-Res_3 comprise 3, 4, and 6 SE-Res blocks, respectively. The utterance-level feature aggregated from average pooling is classified into 10 defined scene classes through two fully-connected layers (FC). All experiments throughout this technical report, including the baseline, are performed using hyper-parameters denoted in Table 2.

4. ASC USING AUDIO TAGGING REPRESENTATION

4.1. Audio tagging system

Audio tagging is the task of recognizing the existence of various sound events that reside in an input audio recording. In the authors'

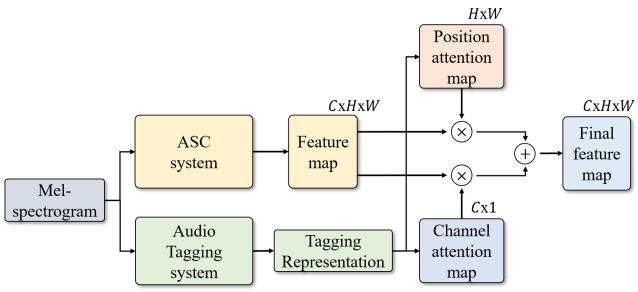

Figure 1: Framework of the tagging representation guided dual attention network for ASC.

previous study [2], it was proposed that using the output layer of an audio tagging system can be helpful for the ASC task. In this technical report, we propose to use the last hidden layer instead of the audio tagging system's output, which we refer to as 'tagging representation'. The audio tagging system used in this study is that of Akiyama *et. al.* [9] which won the DCASE 2019 challenge task 2. Additional details and code of this system are provided in [9][1]

4.2. Tagging representation guided dual attention network for ASC

The proposed overall ASC system using tagging representation is depicted in Figure 1. Mel-spectrogram derived from audio is input in parallel to the ASC system and audio tagging system. The feature map extracted from the ASC system is used as the output of the last SE-Res block of the baseline system. The tagging representation calculated from the audio tagging system is converted into dual attention maps [10]. The positional attention map and the channel attention map perform element-wise multiplication for each channel and position in the feature map, respectively. Each calculated vector is added to each element and converted into a final feature map. The final feature map performs ASC through an average pooling layer and two FC layers.

5. DEEP ARCHITECTURE FOR ASC

To extract discriminative feature for ASC, we explored deep architectures. For training acoustic scenes classifier, we use the light convolutional neural networks (LCNN) framework [3]. LCNN architecture was first introduced for deep face representation with noisy labels. We assume that LCNN architecture with MFM operation that extracts feature maps via a competitive relationship will increase the accuracy of the ASC task. MFM operation focuses on important information through the attention technique without discarding relatively sparse information. To the best of our knowledge, this technical report is the first to adopt a LCNN for the ASC task. We use a LCNN architecture which is similar to that of [11], with some modifications that can be found in Table 3. To emphasize useful information, convolutional block attention module (CBAM) [4] is exploited before pooling layer or batch normalization. CBAM is a simple self-attention module with less computational and parameter overhead. We also utilize specaugment [12] for data augmentation

[1] https://github.com/OsciiArt/Freesound-Audio-Tagging-2019.

Table 3: The LCNN architecture. The numbers in the Output shape column refer to the frame (time), frequency, and number of filters. Conv, MFM, MaxPool and FC indicate convolutional layer, Max-Feature-Map, max pooling layer and fully-connected layer, respectively.

Type	Filter/Stride	Output
Conv_1	7×3 / 1×1	$l \times 124 \times 64$
MFM_1	-	$l \times 124 \times 32$
MaxPool_1	2×2 / 2×2	$(l/2) \times 62 \times 32$
Conv_2a	1×1 / 1×1	$(l/2) \times 62 \times 64$
MFM_2a	-	$(l/2) \times 62 \times 32$
BatchNorm_2a	2×2 / 2×2	$(l/2) \times 62 \times 32$
Conv_2	3×3 / 1×1	$(l/2) \times 62 \times 96$
MFM_2	-	$(l/2) \times 62 \times 48$
CBAM_2	-	$(l/2) \times 62 \times 48$
MaxPool_2	2×2 / 2×2	$(l/4) \times 31 \times 48$
BatchNorm_2	2×2 / 2×2	$(l/4) \times 31 \times 48$
Conv_3a	1×1 / 1×1	$(l/4) \times 31 \times 96$
MFM_3a	-	$(l/4) \times 31 \times 48$
BatchNorm_3a	2×2 / 2×2	$(l/4) \times 31 \times 48$
Conv_3	3×3 / 1×1	$(l/4) \times 31 \times 128$
MFM_3	-	$(l/4) \times 31 \times 64$
CBAM_3	-	$(l/4) \times 31 \times 64$
MaxPool_3	2×2 / 2×2	$(l/8) \times 16 \times 64$
Conv_4a	1×1 / 1×1	$(l/8) \times 16 \times 128$
MFM_4a	-	$(l/8) \times 16 \times 64$
BatchNorm_3a	2×2 / 2×2	$(l/8) \times 16 \times 64$
Conv_4	3×3 / 1×1	$(l/8) \times 16 \times 64$
MFM_4	-	$(l/8) \times 16 \times 32$
CBAM_4	-	$(l/8) \times 16 \times 32$
BatchNorm_4	2×2 / 2×2	$(l/8) \times 16 \times 32$
Conv_5a	1×1 / 1×1	$(l/8) \times 16 \times 64$
MFM_5a	-	$(l/8) \times 16 \times 32$
BatchNorm_5a	2×2 / 2×2	$(l/8) \times 16 \times 32$
Conv_5	3×3 / 1×1	$(l/8) \times 16 \times 64$
MFM_5	-	$(l/8) \times 16 \times 32$
CBAM_5	-	$(l/8) \times 16 \times 32$
MaxPool_5	2×2 / 2×2	$(l/16) \times 8 \times 32$
FC_1	-	160
MFM_FC1	-	80
FC_2	-	10

with mixup, when we train the network without tagging representation. We only use frequency and time masking except time warping as there is little improvement in performance.

6. RESULTS

Following the fold 1 evaluation setup for DCASE2020 task1-a, Table 4 describes class&device-wise classification accuracies of the baseline and the ensemble of two proposed systems. The proposed system demonstrates higher classification accuracies across all nine devices including 6 augmented devices. In terms of each acoustic scene, the proposed system outperformed the baseline in all scenes but Street_traffic in which accuracy decreased from 82.0% to 78.1%. The average of class&device-wise accuracy increased from 65.3% to 71.9% showing 10.1% relative improvement over the baseline.

The results of fold 1 for the four submitted systems are shown

Table 4: Comparison of class&device-wise classification accuracies of the baseline and the proposed system on fold1 test set (baseline/proposed, %). ***Bold*** describes higher accuracy in each scene&device.

	A	B	C	S1	S2	S3	S4	S5	S6	Average
Airport	63.4/**71.9**	69.7/**70.6**	72.4/**78.1**	57.1/**75.0**	51.5/**63.9**	64.0/**71.4**	54.5/**68.8**	61.8/**65.6**	58.8/**67.7**	61.4/**70.3**
Bus	83.8/**90.9**	80.0/**100**	**85.7**/84.2	73.0/73.0	77.1/**81.8**	78.1/**82.9**	**81.0**/76.9	78.8/**85.7**	71.9/**81.3**	78.8/**84.1**
Metro	75.0/**82.4**	**72.7**/70.0	68.9/**71.9**	73.1/**82.1**	61.3/**70.6**	60.6/**76.9**	69.6/**75.0**	68.8/**83.9**	63.7/**72.7**	68.2/**76.2**
Met_sta	60.5/**75.0**	59.4/**75.8**	46.8/**61.8**	60.6/**79.4**	58.8/**61.9**	67.7/**75.0**	56.8/**79.3**	64.1/**74.3**	47.1/**71.0**	58.0/**72.6**
Park	**81.6**/79.4	66.0/**73.7**	**90.0**/83.9	73.6/**77.4**	77.4/**86.7**	72.5/**74.3**	78.4/**84.4**	63.4/**80.0**	68.4/**85.7**	74.6/**80.6**
Pub_squ	66.7/**79.2**	52.4/**56.7**	58.5/**60.0**	61.9/**76.0**	**70.8**/62.1	71.4/**80.0**	**62.5**/59.3	**68.4**/64.0	57.1/**61.3**	63.3/**66.5**
Shop_mall	**59.4**/52.8	47.1/**53.1**	73.1/**74.1**	**57.5**/57.1	41.0/**61.8**	38.6/**52.8**	50.0/**53.1**	58.8/**59.5**	54.5/**57.1**	53.3/**57.9**
Street_pede	**61.3**/60.0	57.6/**68.8**	52.4/**64.0**	50.0/**62.1**	35.1/**66.7**	50.0/**63.3**	37.9/**43.8**	58.1/**59.3**	41.4/**66.7**	49.3/**61.6**
Street_traf	**84.8**/73.3	75.7/**82.9**	**82.4**/76.9	**80.0**/78.4	**82.9**/78.4	**82.9**/76.3	**88.2**/74.4	79.4/**81.6**	**82.1**/80.6	**82.0**/78.1
Tram	72.4/**82.4**	75.0/**78.1**	65.8/**67.6**	57.6/**66.7**	61.3/**74.2**	65.7/**68.3**	69.4/69.4	60.6/**67.6**	51.2/**66.7**	64.3/**71.2**
Average	70.9/**74.7**	65.6/**73.0**	69.6/**72.3**	64.4/**72.7**	61.7/**70.8**	65.2/**72.1**	64.8/**68.4**	66.2/**72.2**	59.6/**71.1**	65.3/**71.9**

Table 5: Results of submitted systems on the fold 1 test set.

System ID	Accuracy(%)
#1	71.9
#2	71.0
#3	70.5
#4	68.8

in Table 5. Note that for actually submission, we self-configured fold 2 through fold 4 and conducted score-sum ensemble. We applied a test time augmentation (TTA) method for extracting embeddings to train support vector machine (SVM) classifier and evaluation phase [13]. Model ensemble is conducted in score-level after training SVM classifier with rbf kernel only or both rbf and sigmoid kernels. Submitted 4 systems are as follows :

1. Ensemble of tagging representation and deep architecture (SVM: rbf, sigmoid)
2. Ensemble system of tagging representation and deep architecture (SVM: rbf)
3. Deep architecture system (SVM: rbf)
4. Tagging representation system (SVM: rbf, sigmoid)

The performance of the submitted four systems on the fold1 test set shown in Table 5, sequentially. All four systems reported higher accuracy than the baseline, and the ensemble system #1 and #2 showed higher performance than the single system #3 and #4.

7. REFERENCES

[1] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in dcase 2020 challenge: generalization across devices and low complexity solutions," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, 2020, submitted. [Online]. Available: https://arxiv.org/abs/2005.14623

[2] J.-w. Jung, H.-j. Shim, J.-h. Kim, S.-b. Kim, and H.-J. Yu, "Acoustic scene classification using audio tagging," *arXiv preprint arXiv:2003.09164*, 2020.

[3] X. Wu, R. He, Z. Sun, and T. Tan, "A light cnn for deep face representation with noisy labels," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2884–2896, 2018.

[4] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–19.

[5] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[6] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[7] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[9] O. Akiyama and J. Sato, "Dcase 2019 task 2: Multitask learning, semi-supervised learning and model ensemble with noisy data for audio tagging," 2019.

[10] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3146–3154.

[11] G. Lavrentyeva, S. Novoselov, A. Tseren, M. Volkova, A. Gorlanov, and A. Kozlov, "Stc antispoofing systems for the asvspoof2019 challenge," *arXiv preprint arXiv:1904.05576*, 2019.

[12] 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," *arXiv preprint arXiv:1904.08779*, 2019.

[13] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," *arXiv preprint arXiv:1806.05622*, 2018.