FEATURE ENHANCEMENT FOR ROBUST ACOUSTIC SCENE CLASSIFICATION WITH DEVICE MISMATCH

Technical Report

Hongwei Song[1], *Hao Yang*[1]

[1] Harbin Institute of Technology,
Department of Computer Science and Technology, Harbin, China,
{songhongwei}@hit.edu.cn, 17S003015@stu.hit.edu.cn

ABSTRACT

This technical report describes our system for DCASE2019 Task1 SubtaskB. We focus on analyzing how device distortions affect the classic log Mel feature, which is the most adopted feature for convolutional neural networks (CNN) based models. We demonstrate mathematically that for log Mel feature, the influence of device distortion shows as an additive constant vector over the log Mel spectrogram. Based on this analysis, we propose to use feature enhancement methods such as spectrogram-wise mean subtraction and median filtering, to remove the additive term of channel distortions. Information loss introduced by the enhancement methods is discussed. We also motivate to use mixup technique to generate virtual samples with various device distortions. Combining the proposed techniques, we rank the second on the public kaggle leaderboard.

Index Terms— Robust acoustic scene classification, device mismatch

1. INTRODUCTION

DCASE2019 Task1b [1] is concerned with the challenge of device mismatch for acoustic scene classification (ASC). The audio was captured using devices with various qualities, such as high-quality microphone, smart phones and cameras.

As in Fig 1, the audio-recording process can be represented by using a classic dynamic system diagram, where the the original signal $s(n)$ is convolved with the system impulse response (IR) $h(n)$ to get the recorded signal $x(n)$, i.e., in the time domain,

$$x(n) = s(n) * h(n) \tag{1}$$

The effects of *device distortions* and *reverberation* are both convolutional, thus they are mingled in $h(n)$. For tasks such as automatic speech recognition, speaker recognition or music processing, these effects are usually categorized as *convolutional noises* and are deemed as distractors to the recognition system.

One interesting property of the *convolutional noises* is that convolutional noise in time domain becomes additive in the log frequency domain, i.e.,

$$\log[X(f)] = \log[S(f)] + \log[H(f)] \tag{2}$$

where $X(f)$, $S(f)$ are the frequency domain representation of the recorded signal and the original signal respectively, and $H(f)$ is the frequency response of $h(n)$. Thus by assuming that the device and recording environment characteristic is stable (or changes slowly),

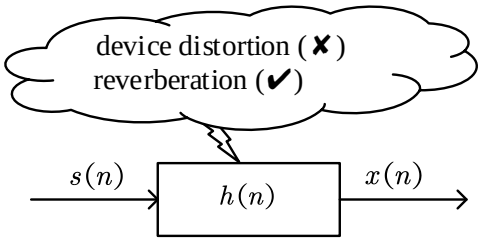

Figure 1: System diagram of the audio recording process.

the *convolutional noises* can be removed by popular techniques such as cepstral mean subtraction (CMS) [2].

However, for recognizing acoustic scenes, the effect of *reverberation* is an important cue for both human [3] and automatic ASC system [4] to infer the spacial size of the environment. Therefore, we want to keep the *reverberation* information while removing the device distortion. And conventional feature-domain channel normalization methods (such as CMS) would be unsatisfactory as it will remove *reverberation* information as well.

In the next section, we will extend our analysis to how device distortion affect the classic log Mel feature and motivate our feature enhancement methods, as well as mixup training.

2. ANALYSIS

2.1. Influence of device distortion on log Mel feature

The feature extraction process for log Mel is shown in in Fig 2. For an input audio $x(n)$, Short-Time Fourier Transform (STFT) power spectrogram $X(f,t)$ is first extracted, where t is the index of frames and f is the index of Fourier Transform points. With (1) and the property of Fourier Transform, it is easy to see that:

$$X(f,t) = S(f,t)H(f) \tag{3}$$

where $S(f,t)$ is the STFT power spectrogram of the original signal, $X(f,t)$ is the STFT power spectrogram of the recorded signal and $H(f)$ is the frequency response corresponding to the effect of device distortion and reverberation.

Triangular Mel-bank is then used to integrate frequency components to generate a auditory-like spectrogram $X(m,t)$, where m

is the index for Mel energies. According to the analysis of [5], the following equation holds,

$$\log[X(m,t)] = \log[S(m,t)] + \log[H(m)] \tag{4}$$

only if frequency response $h(f)$ does not change within each frequency band of the Mel-bank. This implies that to make device distortion and reverberation effect additive in log Mel domain, it's better to use large number of Mel-bank.

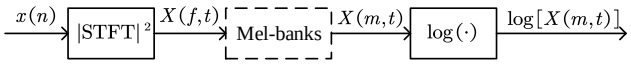

Figure 2: Flow chart of the log Mel feature.

In summary, with large number of Mel-banks, the influence of the *device distortion* shows as an additive constant vector $\log[H(m)]$ over the log Mel feature of the original signal ($\log[S(m,t)]$). And it needs to keep in mind that, the reverberation effect is also mingled in the term $\log[H(m)]$.

3. METHODS

3.1. Spectrogram-wise mean subtraction

To remove the influence of the additive device distortion term, we propose to use spectrogram-wise mean subtraction in the log Mel domain. The mean subtraction technique can be described as:

$$log[X(m,t)] - \mathbf{E}_t[log[X(m,t)]] \tag{5}$$

where $\mathbf{E}_t[\cdot]$ stands for expectation operation over frame t. Substitute (4) into (5), this reduces to:

$$log[S(m,t)] - \mathbf{E}_t[log[S(m,t)]] \tag{6}$$

The additive device distortion term $log[H(m)]$ is removed. Notably, the mean subtraction operation is applied independently for each spectrogram, which is different from the standarization process.

However, the side effect is that the mean of log Mel the original signal $\mathbf{E}_t[log[S(m,t)]]$ is also removed, which loses information. Besides, the reverberation information is also lost.

3.2. Median Filtering

Median filtering has been found to be quite effective in feature pre-processing for ASC [6] [7]. Typically, the median-filtered log Mel spectrogram is subtracted from the original one, thus the background drift is removed emphasizing the sharp spectral changes. The filter size is set to (7, 21) by preliminary experiments, which spans 7 Mel bins and 21 frames.

For our task, median filtering is applied because it helps to alleviate the additive device distortion, especially when the $log[H(m)]$ changes slowly over the frequency (i.e. m) axis.

3.3. Mixup training

Mixup [8] is a simple data-augmentation method which constructs virtual data-label pair $(\tilde{x}, \tilde{y})$ using convex combination of two random data-label pairs (x_i, y_i) and (x_j, y_j):

$$\begin{aligned} \tilde{x} &= \lambda x_i + (1-\lambda) x_j \\ \tilde{y} &= \lambda y_i + (1-\lambda) y_j \end{aligned} \tag{7}$$

where $\lambda \in [0,1]$ is the interpolation coefficient, which is generated randomly from the symmetric Beta distribution:

$$\lambda \sim \mathrm{Beta}(\alpha, \alpha), where\ \alpha \in (0,1] \tag{8}$$

For our system, hyper parameter α is set to 0.3, which is determined by preliminary experiments.

Originally, *mixup* was motivated from the principle of Vicinical Risk Minimization (VRM) [8]. Recently, it has also been successfully applied to ASC to reduce generalization error [9]. However, here we motivate using *mixup* from a different perspective. That is, by mixing up audio recorded from different devices in the log Mel domain, the additive device distortion term ($log[H(m)]$) is also mixed. As a result, the neural net sees samples with various device distortions, therefore the net should be robust with device distortion.

3.4. CNN Model

For the CNN model, we adopt the same model as in [10], which achieve the state-of-the-art performance for ASC without model fusion. The modified Xception model with multi-scale (MS) property is utilized, which we will refer to as MSXception model. Parameters pre-trained from ImageNet dataset [11] is loaded as the initialization for the model. The models are implemented using Pytorch [12]. For fully reproducing the reported results, we have made our code and all the related experiments publicly available at https://github.com/hackerekcah/dcase19_task1_hitsplab.git

4. EXPERIMENTS

4.1. Datasets

Development set of the DCASE2019 Task1 Subtask B [1] is utilized for experiments. No external data is used except that the model is initialized with parameters [11] trained on ImageNet dataset, which boosts performance by a large margin. The default official partition of training and testing fold is adopted. The model trained on training fold is used to evaluate on leaderboard dataset, as well as the challenge evaluation set. Number of segments for each device in the development set is listed in Table 1.

Table 1: Number of audio segments for each device.

device	device_A	device_B	device_C
train	9185	540	540
test	4185	540	540

4.2. Features and global normalization

For input features, the log Mel spectrogram is firstly extracted using librosa library [13] from each audio wave, with a frame length of

Table 2: Systems trained using validation *Metric A*.

System ID	pre-train	global_norm	*mean_sub*	*medfilter*	*mixup*(α)	*A*(%)↑	*B*(%)↑	*C*(%)↑	*B&C*(%)↑	*device_gap*(%)↓
Baseline [1]	–	–	–	–	–	61.9	39.6	43.1	41.4	20.5
A1	false	GMSVN	×	×	×	69.2	45.7	53.5	49.6	19.6
A2	true	GMSVN	×	×	×	76.4	55.2	62.2	58.7	17.7
A3	true	GMS	✓	×	×	67.7	65.2	61.1	63.1	4.6
A4	true	GMS	×	✓	×	71.8	58.1	59.3	58.7	13.1
A5	true	GMSVN	×	×	0.3	77.8	60.6	65.9	63.2	14.6
A6	true	GMS	✓	×	0.3	71.4	69.3	66.9	**68.1**	**3.3**
A7	true	GMS	×	✓	0.3	73.4	60.6	63.3	61.9	11.5
A5 A6 A7	–	–	–	–	–	79.9	69.6	71.7	**70.6**	9.3

Table 3: Systems trained using validation *Metric B*.

System ID	pre-train	global_norm	*mean_sub*	*medfilter*	*mixup*(α)	*A*(%)↑	*B*(%)↑	*C*(%)↑	*B&C*(%)↑	*device_gap*(%)↓
B1	false	GMSVN	×	×	×	68.8	48.0	54.4	51.2	17.6
B2	true	GMSVN	×	×	×	76.0	56.5	60.2	58.3	17.7
B3	true	GMS	✓	×	×	67.8	63.9	61.5	62.7	5.1
B4	true	GMS	×	✓	×	71.6	59.6	61.7	60.6	11.0
B5	true	GMSVN	×	×	0.3	76.7	64.3	67.0	65.6	11.1
B6	true	GMS	✓	×	0.3	69.6	67.2	67.6	**67.4**	**2.2**
B7	true	GMS	×	✓	0.3	74.0	60.2	65.6	62.9	11.1
B5 B6 B7	–	–	–	–	–	79.4	69.1	71.9	**70.5**	8.9

4096, hop size of 1024, and 128 Mel-band energies. Therefore, a feature map of shape (128, 430) is generated for each audio sample.

For global normalization, mean μ and standard variance σ of the log Mel energies over all frames and the whole training (or development) set are calculated and used to normalize both training (or development) and validation (or evaluation) set. Two types of global normalization methods are adopted. For the Global Mean Subtraction and Variance Normalization (GMSVN) method, μ is subtracted from each feature map and divide by σ. While for Global Mean Subtraction (GMS), μ is subtracted from each feature map. GMS mehthod is used whenever we apply the *spectrogram-wise mean subtraction* and *median filtering* preprocessing methods.

Since the original Xception model accepts tree channel (RGB) image, the log Mel feature (normalized and processed by the proposed methods) is repeated three times in the channel axis, resulting in a feature map of shape (3, 128, 430).

4.3. Training protocals

Models are trained using an Adam [14] optimizer with a batch size of 32 and an initial learning rate of 0.001. The accuracy on testing fold is used as validation metric for training scheduling. We decay the learning rate with a factor of 0.5 when the accuracy on testing fold does not improve for 3 consecutive epochs. We train the models for 80 epochs and the model with the highest testing accuracy is saved.

Two types of validation metric on testing fold are utilized to monitor the model during training. For *Metric A*: accuracy over the whole testing fold is monitored (i.e., ignoring the device attribute). For *Metric B*: mean of the accuracy on data from device A and the accuracy on data from device B&C is monitored. Due to the fact that the amount of data from device A is far more than device B&C on the testing fold, *Metric B* tends to select model that biases the accuracy on device B&C.

4.4. Experimental results and discussions

Systems trained using validation *Metric A* are listed in Table 2 and systems trained using validation *Metric B* are listed in Table 3. Each system (row) is described in terms of pre-training condition, global normalization methods used, the inclusion (✓) and exclusion (×) of feature enhancement methods and *mixup*. Device-wise accuracy on testing fold is reported. In particularly, for better demonstrating the performance gap between devices, a new metric (*device_gap*) is defined as the accuracy difference between device A and device B&C.

4.4.1. Ablation study for pre-training

By comparing system pair (A1 , A2) and (B1, B2), it can be seen that pre-training using ImageNet dataset brings significant boost of performance.

4.4.2. Ablation study for mean subtraction

By comparing system pair (A2 , A3) and (A5, A6), as well as (B2 , B3) and (B5, B6), it can be seen that *mean_sub* significantly boosts accuracy on device B&C. According to our previous analysis, this

Table 4: List of submissions.

Submission Label	*System ID*	*Fusion*	LB(%)	Rank
Song_HIT_task1b_1	B5	No	–	–
Song_HIT_task1b_2	A5 A6 A7	Yes	76.5	**2** /28
Song_HIT_task1b_3	B5 B6 B7	Yes	–	–

is explainable since the additive device distortion terms is removed. This eliminates input distribution mismatch, so that model trained on data from device A will perform equally well on data from device B&C after the *mean_sub* pre-processing. Meanwhile, a performance degradation on device A is observed, which according to our analysis, is owing to the loss of *reverberation* information, which is an important cue for discriminating acoustic scenes.

4.4.3. Ablation study for median filtering

By comparing system pair (A2 , A4) and (A5, A7), as well as (B2 , B4) and (B5, B7), the device performance gap is diminished moderately, however the performance on device A is greatly degradated, which according to our analysis, is owing to the dramatic information loss caused by median filtering. This technique should be used only in the fusion system.

4.4.4. Ablation study for mixup

By comparing system pairs (A2, A5), (A3, A6), (A4,A7), as well as (B2, B5), (B3, B6), (B4,B7), it can be seen *mixup* not only boosts performance on all devices, it also diminish performance gap between devices, which accords with our previous analysis.

4.4.5. System fusion

A simple fusion of three systems trained on various feature enhancement methods is investigated. Probability aggregation over three systems is utilized as fusion strategy. It is clear that the fusion system improves performance on all devices.

5. SUBMISSIONS

Three systems are submitted to the challenge website for final evaluation, including one single-model system and two fusion-based systems. Table 4 lists the *Submission Label*s with their corresponding *System ID*s, which can be used for quick referencing system configurations in Table 2 and Table 3.

For single-model system, system B5 is adopted, it provide excellent performance on data from device A and reasonable performance on device B&C. We won't argue this would be the best for our proposed single system, since the final evaluation may be tested on device B&C.

With fusion system Song_HIT_task1b_2, our team (HIT_SPLAB) achieves an accuracy of 76.5% on public leadearboad for DCASE2019 Task1B, which rank the second among twenty eight teams.

Fusion system Song_HIT_task1b_3 uses the same data and model as Song_HIT_task1b_2, but is trained by monitoring the validation *Metric B*. Since the final evaluation metric will be based on accuracy on device B&C, we expect this system may outperform Song_HIT_task1b_2 in this case.

6. CONCLUSIONS

In this paper, we showed that for log Mel feature, the influence of device distortion shows as an additive constant vector over the log Mel spectrogram. We demonstrate that mean subtraction can eliminate device distortion, but it also brings information loss. Median filtering is also effective for diminishing performance gap between devices, but it cause drastic information loss and should only be used in fusion systems. Most importantly, mixup not only boosts performance on all devices, but also helps to diminish device mismatch problem. The analysis and the proposed methods are mainly based on the classic log Mel feature, which can be easily integrated with powerful CNN models. For future research, model-based domain adaptation methods may be combined with our enhanced feature to boost performance.

7. REFERENCES

[1] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," in *Proc. DCASE2018 Workshop*, November 2018, pp. 9–13.

[2] A. E. Rosenberg, C.-H. Lee, and F. K. Soong, "Cepstral channel normalization techniques for hmm-based speaker verification," in *Third International Conference on Spoken Language Processing*, 1994.

[3] J. Traer and J. H. McDermott, "Statistics of natural reverberation enable perceptual separation of sound and space," *Proceedings of the National Academy of Sciences*, vol. 113, no. 48, pp. E7856–E7865, 2016.

[4] M. Marković and J. Geiger, "Reverberation-based feature extraction for acoustic scene classification," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 781–785.

[5] C. Avendano and H. Hermansky, "On the effects of short-term spectrum smoothing in channel normalization," *IEEE transactions on speech and audio processing*, vol. 5, no. 4, pp. 372–374, 1997.

[6] Y. Han, J. Park, and K. Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," *the Detection and Classification of Acoustic Scenes and Events (DCASE)*, pp. 1–5, 2017.

[7] Y. Wu and T. Lee, "Enhancing sound texture in cnn-based acoustic scene classification," *arXiv preprint arXiv:1901.01502*, 2019.

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

[9] K. Xu, D. Feng, H. Mi, B. Zhu, D. Wang, L. Zhang, H. Cai, and S. Liu, "Mixup-based acoustic scene classification using multi-channel convolutional neural network," in *Pacific Rim Conference on Multimedia*. Springer, 2018, pp. 14–23.

[10] Y. Liping, C. Xinxing, and T. Lianjie, "Acoustic scene classification using multi-scale features," in *Proc. DCASE2018 Workshop*, November 2018, pp. 29–33.

[11] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[12] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[13] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, 2015, pp. 18–25.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.