SEMI-SUPERVISED SOUND EVENT DETECTION BASED ON MEAN TEACHER WITH POWER POOLING AND DATA AUGMENTATION

Technical Report

Yuzhuo Liu[1,2]*, Chengxin Chen*[1,2]*, Jianzhong Kuang*[1,2]*, Pengyuan Zhang*[1,2]

[1] Institute of Acoustics, Key Laboratory of Speech Acoustics & Content Understanding, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
{liuyuzhuo, chenchengxin, kuangjianzhong, zhangpengyuan}@hccl.ioa.ac.cn

ABSTRACT

In this technical report, we describe the details of the system submitted to DCASE2020 task4: sound event detection (SED) and separation in domestic environments. We mainly focus on the scenario that recognizes sound events without source separation. The training set includes synthetic strongly labeled data, weakly labeled data and unlabeled data. For training SED models with weak labeling, a power pooling function is introduced to generate clip-level predictions from frame-level ones. Additionally, three traditional data augmentation approaches are applied on all data. We also ensemble models with different strategies. Our best system finally achieves an F1 of 49.55% on the validation set.

***Index Terms*—** DCASE2020, sound event detection, power pooling, data augmentation

1. INTRODUCTION

DCASE2020 task4 is an extension of DCASE2018 task4 and DCASE2019 task4. All Task4 challenges pay attention to semi-supervised sound event detection (SED) in domestic environments. DCASE2018 task4 takes only weakly labeled data (without timesteps) and unlabeled data as training data. DCASE2019 task4 adds synthetic strongly labeled data and discuss the necessity of this behaviour. DCASE2020 task4 encourages to detect sound events jointly with sound separation, and divides the task into three scenarios:

1.Train a SED system without source separation preprocessing.

2.Use a source separation algorithm together with SED.

3.Work only on source separation and use the baseline SED system.

The last two scenarios aim to investigate the improvement of SED systems with source separation. Another scientific question that this task concerns is how to optimally exploit synthetic data. Please refer [1] for other detailed information about the challenge.

This technical report focuses on the first scenario and mainly makes two contributions. Firstly, to exploit synthetic data, we apply various traditional data augmentation methods on the training data. Some of them show positive impact on DCASE2020 task4. Secondly, SED with weak labeling is treated as multiple instance learning (MIL) problem. Because an event happens in a clip if and only if the event actives in at least one frame. The strong (frame-level) predictions need to be aggregated with a pooling function to produce weak (clip-level) ones of each clip. A power pooling function [2] is introduced and illustrates a good performance.

2. PROPOSED METHODS

In this section, we describe our systems in 6 parts. Our systems are modified based on the mean teacher model for DCASE2018 task4 [3].

2.1. Data augmentation

Four different audio data augmentations (deformations) are applied on all training dataset [4]. Each deformation was performed directly to the audio signal before feature extraction. The deformations and resulting augmentation sets are described below:

1.Audio time stretching (ATS): slow down or speed up the audio sample while keeping the pitch unchanged. We experimented with two strategies: (1) each sample was time stretched by 2 factors: 0.8, 1.2. (2) each sample was time stretched by a factor which was sampled uniform from 0.8 to 1.2. Both methods failed to improve the performance of our systems. The reason might be that the variety of the event length resulted in mismatching with the window sizes of the median filter computed from official synthetic data.

2.Audio pitch shifting (APS): raise or lower the pitch of the sample while keeping the duration unchanged. We experimented with two strategies: (1) each sample was pitch shifted by 2 values (in semitones): +2, -2. (2) each sample was pitch shifted by a value which is sampled uniform from -2 to +2. The results showed that the former strategy (fixed pitch shift) is more effective.

3.Audio time rolling (ATR): shift the time of the sample while keeping the duration and pitch unchanged, also called audio time shifting. Each sample was time rolled by a parameter which was a product of the audio duration and a factor. The factor was chosen randomly from 0.1 to 0.9.

4.Dynamic range compression (DRC): compress the dynamic range of the sample using the parameter *speech* taken from the Dolby E standard [5]. Consider that the speech events account for the highest proportion of the datasets.

Some of the deformations (APS, ATS, DRC) are performed using the MUDA toolkit [6].

2.2. Audio preprocessing

The sampling rate of clips is 44100 Hz. We resampled the clips at 16,000 HZ, and converted multi-channel ones to single-channel. The 128-dimensional log mel-spectrogram was extracted by 743-window and 132-hop at each frame. The 10-second audio clips should be convert to a 1209-frames float data as the feature.

Table 1: The clip-level and frame-level gradient direction of linear pooling. y_f and y_c are frame-level and clip-level predictions.

Label	Clip-level	Condition	Frame-level
positive ($t = 1$)	$y_c \to 1$	$y_f > y_c/2$	$y_f \to 1$
		$y_f < y_c/2$	$y_f \to 0$
negative ($t = 0$)	$y_c \to 0$	$y_f > y_c/2$	$y_f \to y_c/2$
		$y_f < y_c/2$	$y_f \to y_c/2$

2.3. Model architecture

Our SED model was modified based on the structure proposed in [3]. The log mel-spectrogram was fed to a mean teacher [7] model. Teacher and student model share the same structure of CRNN. The weights of the student model are updated with gradient descent, and the teacher model weights are updated as an exponential moving average (EMA) of the student weights. The CRNN model consists of 7 2-D convolutional blocks, 2 bi-directional gated recurrent units (BiGRU) and 1 dense layer. The number of filters and pooling sizes are [16,32,64,128,128,128,128] and [[2,2],[2,2],[1,2],[1,2],[1,2],[1,2],[1,2]] respectively . Context Gating (CG) [8], Gated Linear Units (GLU) [9] are used as the activation in different models. The training epoch was set to 200.

2.4. Pooling function

A power pooling was adopted to produce clip-level predictions from frame-level ones. Linear pooling function [10], whose formula and gradient can be written as:

$$y_c = \frac{\sum_i y_f(i) \times y_f(i)}{\sum_i y_f(i)}, \tag{1}$$

$$\frac{\partial y_c}{\partial y_f(i)} = \frac{2y_f(i) - y_c}{\sum_j y_f(j)}, \tag{2}$$

has been confirmed to work best among classic five pooling functions for frame-level classification [11], since the clip-level and frame-level gradient directions change as Table 1. For positive recordings ($t = 1$), linear pooling function leads larger y_f to be boosted.

Limited by the form of linear function, the threshold of larger y_f is defined as $y_c/2$. However, the threshold is supposed to be adjusted dynamically according to the value of y_c and the number of positive frame-level samples. This issue can be addressed by applying a power pooling function:

$$y_c = \frac{\sum_i y_f(i) \times y_f^n(i)}{\sum_i y_f^n(i)}, \tag{3}$$

and the gradient can be written as:

$$\frac{\partial y_c}{\partial y_f(i)} = \frac{(n+1) \times y_f^n(i) - n \times y_f^{n-1}(i) \times y_c}{\sum_j y_f^n(j)}. \tag{4}$$

Treating n as a free parameter to be learned along-side the model parameters allows eq.(3) to automatically adapt to and interpolate between separate pooling functions. For instance, when $n = 0$, eq.(3) reduces to mean pooling. When $n = 1$, eq.(3) simplifies to linear pooling. When $n \to \infty$, eq.(3) approaches the max aggregation.

Table 2: The amount of events, mean durations (in s), median durations (in s) and window sizes of the median filter (in s) for synthetic data.

Class	Amout	Mean	Median	Window_sizes
Vacuum_cleaner	342	5.84	6.04	2.92
Frying	201	5.56	5.51	2.82
Electric_shaver_toothbrush	334	4.64	4.21	2.24
Running_water	271	4.08	3.65	1.98
Blender	414	2.52	1.51	1.20
Speech	2625	1.13	0.90	0.49
Cat	783	1.10	0.88	0.48
Alarm_bell_ringing	532	1.01	0.33	0.21
Dog	858	0.91	0.45	0.26
Dishes	1115	0.57	0.41	0.20
total_segments	7475	/	/	/

2.5. Median filter

We used an adaptive post-processing [12] where the window size of median filters for each event category c is obtained with the formula

$$Win_c = duration_c \times \beta_c, \tag{5}$$

For some event categories such as *Alarm/bell/rining*, the variance of the duration is large. Therefore, we adopted the median duration instead of mean duration of each sound event class as $duration_c$. β firstly took 0.55. Then, we adjusted the window sizes manually. The amount of events, mean durations, median durations and window sizes of the median filter for synthetic data are in Table 2.

2.6. System fusion

We fuse the posteriors (after median filter) of models with different activation (CG and GLU) and data augmentations. The final results are generated from the weighted-average predictions of single models. A grid search is utilized to determine a group of optimal weights for various event classes.

3. EXPERIMENTS

In this section, we first describe the information about the dataset of DCASE2020 task4. Then, the results of our systems are shown.

3.1. Dataset

The DCASE2020 task4 dataset can be divided into four subsets, including training sets (synthetic strongly labeled: 2,595 clips, weakly labeled: 1,578 clips, unlabeled: 14,412 clips) and validation set (1,168 clips). The duration of majority audio clips is 10 seconds, and multiple audio events may occur at the same time. We combined the original data with the beneficial augmented data respectively to produce 3 new datasets.

3.2. Results

According to the data augmentations and activation, 6 single systems are trained. The F1 scores of these systems are illustrated in Table 3.

Our 4 submissions are the fusion of above 6 models. The ensemble strategy is as follows:

System1 (S1): the result is the weighted-average of the results generated from model M_{aps_cg}, M_{atr_cg} and M_{drc_cg}.

Table 3: The event-based F1 (%) of 6 single models.

Model	F1
Mean-Teacher-APS-CG (M_{aps_cg})	42.98
Mean-Teacher-ATR-CG (M_{atr_cg})	41.89
Mean-Teacher-DRC-CG (M_{drc_cg})	41.38
Mean-Teacher-APS-GLU (M_{aps_glu})	36.35
Mean-Teacher-ATR-GLU (M_{atr_glu})	38.12
Mean-Teacher-DRC-GLU (M_{drc_glu})	39.19

Table 4: The event-based F1 (%) of submitted four systems.

Class/Model	S1	S2	S3	S4
Vacuum_cleaner	67.52	72.29	56.98	67.06
Frying	54.02	54.02	51.65	51.93
Electric_shaver_toothbrush	47.27	58.57	48.28	52.86
Running_water	39.55	39.55	33.60	37.04
Blender	54.02	54.02	49.69	52.22
Speech	58.42	58.66	57.85	58.74
Cat	45.57	50.27	47.31	46.32
Alarm_bell_ringing	54.10	54.10	47.15	50.31
Dog	26.46	27.06	26.20	25.95
Dishes	26.97	26.97	24.37	25.93
total	**47.39**	**49.55**	**44.31**	**46.84**

System2 (S2) and System3 (S3): These two results are the weighted-average of the results of all 6 models. Different weights are applied in two systems.

System4 (S4): The result is the average of results of S2 and S3.

Table 4 shows the F1 scores of submitted models. Also, the result for each sound event class is available. The performance reaches 47.39%, 49.55%, 44.31%, 46.84%, which significantly improves the official baseline (34.8%) [13]. Besides, it is noted that the results of shortest two sound events (*Dog*, *Dishes*) are consistently low. This issue may address if the parameter β in 5 is relatively smaller for the two events.

4. CONCLUSION

In this technical report, we propose a modified mean teacher model for DCASE2020 task4. After applying three data augmentation methods, the power pooling function and various ensemble strategies, our best system achieves an F1 of 49.55% on the validation dataset of DCASE2020 task4.

5. REFERENCES

[1] http://dcase.community/challenge2020/.

[2] Y. Liu, H. Chen, and P. Zhang, "Power pooling operators and confidence learning for semi-supervised sound event detection," 2020.

[3] L. J., K., "Mean teacher convolution system for dcase 2018 task 4," in *DCASE2018 Challenge*, 2018.

[4] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, pp. 279–283, 2017.

[5] D. E. Bitstreams, "Standards and practices for authoring dolby digital and dolby e bitstreams," 2002.

[6] B. McFee, E. J. Humphrey, and J. P. Bello, "A software framework for musical data augmentation," in *ISMIR*, 2015.

[7] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," pp. 1195–1204, 2017.

[8] J. S. Antoine Miech, Ivan Laptev, "Learnable pooling with context gating for video classification," 2017.

[9] Y. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," 12 2016.

[10] D. A., V. T., H., and W. J., C., "Deep learning for dcase2017 challenge," in *DCASE2017 Challenge*, 2017.

[11] Y. Wang, J. Li, and F. Metze, "A comparison of five multiple instance learning pooling functions for sound event detection with weak labeling," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 31–35.

[12] L. Lin and X. Wang, "Guided learning convolution system for dcase 2019 task 4," Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, Tech. Rep., June 2019.

[13] D.-P. L. and P. C., "Mean teacher with data augmentation for dcase 2019 task 4," in *DCASE2019 Challenge*, 2019.