GUIDED LEARNING CONVOLUTION SYSTEM FOR DCASE 2019 TASK 4

Technical Report

Liwei Lin[1,2], *Xiangdong Wang*[1], *Hong Liu*[1], *YueLiang Qian*[1],

[1]Bejing Key Laboratory of Mobile Computing and Pervasive Device,
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
{linliwei17g, xdwang, hliu, ylqian}@ict.ac.cn

ABSTRACT

In this technical report, we describe in detail the system we submitted to DCASE2019 task 4: sound event detection (SED) in domestic environments. We employ a convolutional neural network (CNN) with an embedding-level attention pooling module to carry out weakly-supvised learning. By considering the interference caused by the co-occurrence of multiple events in the unbalanced dataset, we combine the model with the disentangled feature. To take advantage of the unlabeled data, we adopt guided learning for semi-supervised learning. A group of median filters with adaptive window sizes is utilized in post-processing of frame-level probabilities output by the model. We also analyze the effect of the synthetic data and finally achieve an F-measure of 45.43% on the validation set.

Index Terms— Guided learning, disentangled feature, weakly supervised learning, semi-supervised learning, attention

1. INTRODUCTION

DCASE2019 task 4 is the follow-up to DCASE2018 task 4, which aims at exploring the possibility of the large-scale detection of sound events using weakly labeled data (without timestamps) and unlabeled data. Different from DCASE2018 task 4, DCASE2019 task 4 introduces an additional strongly annotated synthetic training set.

Sound event detection (SED) consists in recognizing the presence of sound events in the segment of audio and detecting their onset as well as offset. Due to the high cost of manually labeling data, it is essential to efficiently utilize weakly-labeled data and unlabeled data. Simultaneously, the different physical characteristics of events (such as different duration) and the unbalance of the available training set also increases the difficulty of polyphonic SED in domestic environments. For DCASE2019 task4, there are 5 issues to be resolved:

1) How to learn efficiently with weakly-labeled data?
2) How to learn efficiently with unbalanced training set?
3) How to combine weakly-supervised learning with semi-supervised learning efficiently using weakly-labeled data and unlabeled data?
4) Does the strongly annotated synthetic training set help?
5) How to design a better post-processing method to detect more accurate boundaries according to the characteristics of each event category?

In this technical report, we present a system to solve all these five issues. For issue 1 and 2, we utilize convolutional neural network (CNN) with the embedding-level attention pooling module and disentangled feature [1] to solve them. For issue 3, we adopt a semi-supervised learning method named Guided Learning [2]. For issue 4, according to varied duration of different event categories, we employ a group of median filters with adaptive window sizes in the post-processing of output probabilities of the model. For issue 5, we simply regard the strongly annotated synthetic training set as a weakly annotated training set and conduct a series of ablation experiments to explore its effects on weakly-supervised learning and unsupervised learning separately.

2. METHOD

In this section, we discuss the solution for issue 1 in Section 2.1, solution for issue 2 in Section 2.2 and solution for issue 3 in Section 2.3.

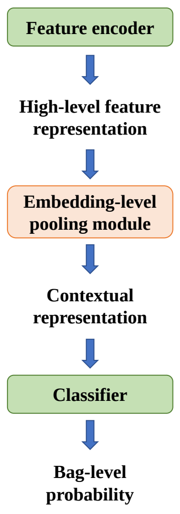
2.1. A CNN model with the embedding-level attention pooling module

As shown in Figure 1a, the model we employ comprises 3 parts: a feature encoder, an embedding-level attention pooling module and a classifier. The feature encoder encodes the input feature of an audio clip into high-level feature representations. Assuming that there are C event categories to detect, then the embedding-level attention pooling module integrates these high-level feature representations into C contextual representations. Eventually, the clip-level probabilities can be obtained by passing this C contextual representations through the classifier.

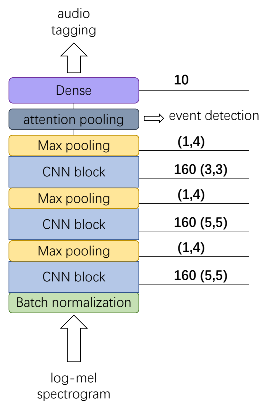
As shown in Figure 1b, the feature encoder we employs is composed of a Batch normalization layer [3], 3 Max pooling layers and 3 CNN blocks, each of which consists of a CNN layer, a Batch normalization layer and a ReLU activation layer as shown in Figure 1c. And the classifier for each contextual representation is a fully-connected layer with a Sigmoid activation layer.

The ability of this model to carry out weakly-supervised learning attributes to its embedding-level attention pooling module. Let $\mathbf{x} = \{x_1, ..., x_T\}$ be the high-level feature representations generated by the feature encoder and $\mathbf{y} = \{y_1, ..., y_C\}$ ($y_c \in \{0, 1\}$) be the groundtruth, where T denotes the number of total frames of the high-level feature representations.

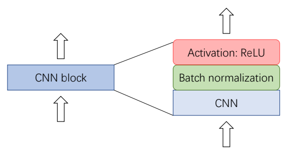
Then for each category c, the embedding-level attention pooling gives different weights $\mathbf{a_c} = \{a_{c1}, ..., a_{cT}\}$ to the corresponding x_t in $\mathbf{x}$. Then the contextual representation $\mathbf{h} = \{h_1, h_2, ..., h_C\}$ can


Feature encoder
High-level feature representation
Embedding-level pooling module
Contextual representation
Classifier
Bag-level probability

(a) Model framework



(b) Detailed model architecture



(c) CNN block

Figure 1: CNN model with the embedding-level attention pooling module.

be obtained by the following way:

$$h_c = \sum_t a_{ct} \cdot x_t \tag{1}$$

Such an $\mathbf{a_c}$ enables the model to treat each frame differently. Important frame x_t in $\mathbf{x}$ with larger a_{ct} contributes more to h_c. The embedding-level attention pooling module generates $\mathbf{a_c}$ by the following way:

$$a_{ct} = \frac{\exp\left((w_c^T x_t + b_c)/d\right)}{\sum_k \exp\left((w_c^T x_k + b_c)/d\right)} \tag{2}$$

where d is equal with the dimension of $\mathbf{x}$, w_c^T is a trainable vector, and b_c is the trainable bias.

importantly, a_{ct} possess the ability to indicate key frames of an audio and is able to generate frame-level probabilities as explained in [1]:

$$\hat{p}\left(y_c \mid x_t\right) = \sigma\left(w_c^T x_t + b_c\right) \tag{3}$$

where σ is Sigmoid function.

Assuming that $\hat{\mathbf{P}}\left(y_c \mid \mathbf{x}\right)$ is the clip-level probabilities for event category c, then the clip-level prediction is:

$$\phi_{\mathbf{c}}(\mathbf{x}) = \begin{cases} 1, & \hat{\mathbf{P}}(1 \mid \mathbf{x}) \geq \alpha \\ 0, & otherwise \end{cases} \tag{4}$$

The frame-level prediction is:

$$\varphi_{\mathbf{c}}(\mathbf{x}, t) = \begin{cases} 1, & \hat{p}(1 \mid x_t) \cdot \phi_{\mathbf{c}}(\mathbf{x}) \geq \alpha \\ 0, & otherwise \end{cases} \tag{5}$$

Without loss of generality, we set $\alpha = 0.5$ in our work.

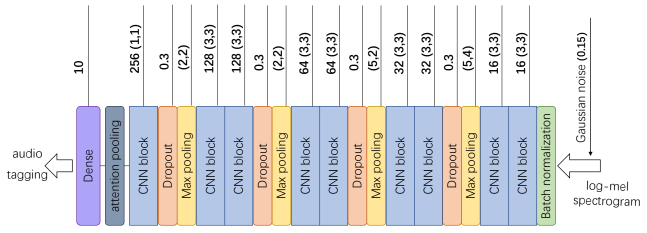


Figure 2: The model architecture of the PT-model.

2.2. Disentangled feature

We take disentangled feature (DF) [1], which re-models the high-level feature subspace of each event category according to the prior information without pre-training, to mitigate the effect of the interference caused by the co-occurrence multiple events.

Assuming that $\chi^{\mathbf{d}}$ ($\mathbf{x} \subset \chi^{\mathbf{d}}$) is a d-dimensional space generated by the feature encoder and $ß = \{e_1, e_2, ..., e_d\}$ is an orthogonal basis of $\chi^{\mathbf{d}}$ where the element of e_i in i^{th} dimensional is 1. DF selects specific bases of $\chi^{\mathbf{d}}$ to construct a specific subspace for each category and the basis of the re-modeled feature space $\chi_c^{'}$ of category c is

$$ß_c^{'} = \{e_1, e_2, ..., e_{k_c}\} \tag{6}$$

$$k_c = \lceil((1-m) \cdot f_c + m) \cdot d\rceil \tag{7}$$

$$f_c = \sum_i^C \frac{r_i \cdot N_{ci}}{R} \tag{8}$$

$$R = \max_c \sum_{i=1}^C r_i \cdot N_{ci} \tag{9}$$

where m is a constant to avoid too-small k_c and N_{ci} is the number of clips containing i categories including category c in the training set. The constant coefficient r_i denotes the importance these clips:

$$r_i = \begin{cases} 1, & i = 1 \\ 0, & otherwise \end{cases} \tag{10}$$

2.3. Guided learning with a more professional teacher

To combine weakly-supervised learning with semi-supervised learning, we utilize Guide Learning (GL) proposed in [2] with a more professional teacher model (PT-model) to guide a more promising student model (PS-model).

The architecture of the PS-model is consistent with the model described in the former two sections and we show that of the PT-model in Figure 2. The CNN feature encoder of the PT-model is considered to be better designed than the PS-model on the audio tagging performance with larger sequential sampling size and less trainable parameters. This is because that the larger sequential sampling size allows the CNN feature encoder of the PT-model to have a larger receptive field followed by better exploitation of contextual information.

However, the larger sequential sampling size also disables the PT-model to see finer information due to the compress of sequential information. Therefore, the PS-model is designed with smaller sequential sampling size to see finer information and then achieves better frame-level prediction.

Algorithm 1 Guided learning pseudocode.

```
Require: $x_k$ = training input with index $k$
Require: $L$ = set of weakly-labeled training input
Require: $U$ = set of unlabeled training input
Require: $y_k$ = label of weakly-labeled input $x_k \in L$
Require: $S_\theta(x)$ = neural network of the PS-model with trainable parameters $\theta$
Require: $T_{\theta'}(x)$ = neural network of the PT-model model with trainable parameters $\theta'$
Require: $g(x)$ = stochastic input augmentation function
Require: $J(t, z)$ = loss function
Require: $\phi(z)$ = prediction generation function
Ensure: $\theta, \theta'$
  for $i = 1 \rightarrow num\_epoches$ do
    if $i > start\_epoch$ then
      $a \leftarrow 1 - \gamma^{i-start\_epoch}$    ▷ calculate the weight of unsupervised loss of the PT-model
    else
      $a \leftarrow 0$
    end if
    for each minibatch ß do
      $s_k \leftarrow S_\theta(x_k \in ß)$    ▷ the coarse-level predicted probability of the PS-model
      $t_k \leftarrow T_{\theta'}(g(x_k) \in ß)$    ▷ the coarse-level predicted probability of the PT-model
      $\tilde{s_k} \leftarrow \phi(s_k)$    ▷ convert the predicted probability into 0-1 prediction
      $\tilde{t_k} \leftarrow \phi(t_k)$
      if $x_k \in L$ then
        $loss \leftarrow \frac{1}{|ß|}\left\{\sum_{x_k \in J \cap ß}\left[J(y_k, s_k) + J(y_k, t_k)\right]\right\}$    ▷ supervised loss
      end if
      if $x_k \in U$ then
        $loss \leftarrow \frac{1}{|ß|}\left\{\sum_{x_k \in U \cap ß}\left[J(\tilde{t_k}, s_k) + a \cdot J(\tilde{s_k}, t_k)\right]\right\}$    ▷ unsupervised loss
      end if
      update $\theta, \theta'$    ▷ update network parameters
    end for
  end for
```

This gap between their ability makes it possible to optimize the PS-model with the guide of the PT-model using unlabeled data. As shown in Algorithm 1, an end-to-end process is employed to train these two models.

3. EXPERIMENTS

3.1. DCASE 2019 Task 4 Dataset

The dataset of DCASE2019 task 4 is divided into 4 subsets: the weakly annotated training set, the unlabeled training set, the strongly annotated validation set and the strongly annotated synthetic training set. Each 10-second audio clip in the dataset contains one or more (or None) of 10 events. We integrate the weakly annotated training set, the unlabeled training set and the strongly annotated synthetic training set (actually we only use weakly labels during training) into a training set and take the validation set as our validation set.

Table 1: The dimension of the disentangled feature when $m = 0.04$ and the window sizes of the median filters when $\beta = \frac{1}{3}$.

Event	DF dimension	Window size (frame)
Alarm/bell/ringing	137	17
Blender	94	42
Cat	134	17
Dishes	69	9
Dog	132	16
Electric shaver/toothbrush	76	74
Frying	34	85
Running water	160	64
Speech	30	18
Vacuum cleaner	113	87

3.2. Feature exaction

We produce 64 log mel-bank magnitudes which are extracted from 40 ms frames with 50% overlap ($n_{FFT} = 2048$) using librosa package [4]. All the 10-second audio clips are extracted to feature vectors with 500 frames.

3.3. Model architecture

The constant factor d_c for the embedding-level attention pooling module is the same as the dimension of disentangled feature for event category c and we take $n = 1, m = 0.04$ for disentangled feature. The dimension of the disentangled feature per category is shown in Table 1. The PS-model has about 2.6 times the number of trainable parameters as the PT-model. The start epoch for GL is set to 5. The PS-model with only weakly-supervised learning is named ATP-DF and the co-teaching of the PS-model and the PT-model is named GL-α-PT in the performance report, where α is a hyper-parameter for GL discussed in Algorithm 1.

3.4. Adaptive post-processing

The median filter is utilized for post-processing. Instead of determining the window size of the median filter empirically, we adopt a group of median filters with adaptive window sizes for different event categories by the following formulation based on the varying length of different event categories in real life:

$$S_{win} = duration_{ave} \cdot \beta \tag{11}$$

We take $\beta = \frac{1}{3}$ in our experiments and the adaptive window sizes for different event categories are shown in Table 1. All the frame-level probabilities output by the network are smoothed by a group of median filters with these adaptive window sizes. After smoothed, the probabilities are converted into the 0-1 prediction with a threshold of 0.5. Then the operation of smoothing is repeated again on the final frame-level prediction.

3.5. Training

The Adam optimizer [5] with learning rate of 0.0018 and mini-batch of 64 10-second patches is utilized to train models in the experiments. The learning rate is reduced by 20% per 10 epochs. If there is no more improvement on clip-level macro F_1 within 20 epochs, the training will early stop and the model with the best performance will be kept for prediction. All the experiments are repeated 30 times and we take the average result as the final result.

Table 2: The performance of models

Model	**Macro F_1 (%)** Event-based	Segment-based
baseline	23.7	55.2
without the synthetic training set		
ATP-DF	25.95 ± 3.22	56.82 ± 1.34
GL-1-PT	35.19 ± 3.86	61.14 ± 3.14
GL-0.996-PT	$\mathbf{36.50 \pm 3.71}$	$\mathbf{62.03 \pm 3.25}$
GL-0.99-PT	36.21 ± 4.63	61.25 ± 2.77
GL-0.98-PT	33.78 ± 2.95	57.54 ± 3.42
with the synthetic training set		
ATP-DF	21.65 ± 2.55	57.02 ± 1.93
GL-1-PT	41.03 ± 2.98	65.58 ± 2.84
GL-0.996-PT	42.02 ± 3.29	$\mathbf{66.62 \pm 1.82}$
GL-0.99-PT	$\mathbf{42.32 \pm 2.21}$	65.78 ± 2.63
GL-0.98-PT	41.16 ± 2.42	63.89 ± 2.20

Table 3: The performance of the models (with the synthetic training set) without disentangled feature ($m = 1$).

Model	**Macro F_1 (%)** Event-based	Segment-based
ATP	21.25 ± 2.13	56.22 ± 2.52
GL-1-PT	38.75 ± 2.42	65.23 ± 3.44
GL-0.996-PT	$\mathbf{40.49 \pm 2.30}$	$\mathbf{66.16 \pm 1.50}$
GL-0.99-PT	40.03 ± 3.38	65.14 ± 2.12
GL-0.98-PT	39.52 ± 3.27	63.46 ± 2.33

3.6. Results

As shown in Table 2, GL-0.99-PT (with synthetic set) achieves the best average performance on event-based macro F_1. As shown in Table 5, the ensemble of the models (GL-0.99-PT) from top2 to top6 achieves the best performance, improving the performance by 21.73 percentage points from the baseline. We submitted the top1 model, the top2 model, the ensemble of the models from top1 to top6 and the ensemble of the models from top2 to top6 to the challenge.

3.6.1. The effect of semi-supervised learning

As shown in Table 2, all the models with semi-supervised learning outperform those only with weakly-supervised learning significantly and the model with the best average performance improves the performance by 20.67 percentage points from the weakly-supervised only method.

3.6.2. The effect of disentangled feature

As shown in Table 3, the performance of all the models without disentangled feature is poorer than those which has. The disentangled feature improves the event-based macro F_1 performance about 1-3 percentage points.

3.6.3. The effect of the adaptive post-processing

As shown in Table 4, instead of using a group of median filters with adaptive window sizes, we pick a fixed window size of 27 for the median filter, which is a relatively superior value empirically. In this case, the performance of all the models (with the synthetic training set) is relatively poor, which demonstrates the effectiveness of the adaptive post-processing.

Table 4: The performance of the models (with the synthetic training set) smoothed by the median filter with a fixed window size of 27.

Model	**Macro F_1 (%)** Event-based	Segment-based
ATP-DF	20.87 ± 2.04	56.95 ± 1.88
GL-1-PT	37.46 ± 3.72	65.07 ± 2.48
GL-0.996-PT	38.58 ± 3.98	$\mathbf{65.99 \pm 2.24}$
GL-0.99-PT	$\mathbf{38.99 \pm 2.15}$	65.21 ± 2.31
GL-0.98-PT	37.64 ± 2.58	63.19 ± 2.12

Table 5: The performance of top1 to top2 and the ensemble of models.

Model	**Macro F_1 (%)** Event-based	Segment-based
Top1	44.47	66.74
Top2	44.02	67.07
Ensemble (Top1-6)	45.28	**69.06**
Ensemble (Top2-6)	**45.43**	69.02

3.6.4. Does the synthetic training set help?

As shown in Table 2, when learning only with weakly labeled data, the synthetic training set not only does not help improve the results but also brings negative effects. But when combining weakly-supervised learning with semi-supervised learning, the synthetic training set contributes a lot so that the performance is raised by about 5-8 percentage points. We argue that the model tends to be overfitting in the synthetic training set and have difficulty in recognizing the audio clips from the real-life recording since the number of audio clips in the synthetic training set is almost 1.3 times as much as that in the weakly annotated training set. However, the large scale of unlabeled data complements this weakness and enable the synthetic training set to play a positive role during training.

4. CONCLUSION

In this technical report, we present a system for DCASE2019 task 4. We release the implement to reproduce our system at https://github.com/Kikyo-16/Sound_event_detection. We utilized a CNN model with an embedding-level attention pooling module to carry out weakly-supervised learning and guided learning to carry out semi-supervised learning. The disentangled feature is employed to raise the performance of the model by reducing the interference caused by the co-occurrence of multiple events in the unbalanced data set. In addition, the adaptive post-processing is proposed to get more accurate detection boundaries. As a result, we achieve 45.43%on the validation set, improving the performance by 21.73% from the baseline.

5. ACKNOWLEDGMENT

This work is partly supported by Beijing Natural Science Foundation (4172058).

6. REFERENCES

[1] L. Lin, X. Wang, H. Liu, and Y. Qian, “Specialized decision surface and disentangled feature for weakly-supervised polyphonic sound event detection,” *ArXiv*, vol. abs/1905.10091, 2019.

[2] L. Lin, X. Wang, H. Liu, and Y. Qian, “Guided Learning for the combination of weakly-supervised and semi-supervised learning,” *arXiv e-prints*, p. arXiv:1906.02517, Jun 2019.

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

[4] Brian McFee, Colin Raffel, Dawen Liang, Daniel P.W. Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto, “librosa: Audio and Music Signal Analysis in Python,” in *Proceedings of the 14th Python in Science Conference*, Kathryn Huff and James Bergstra, Eds., 2015, pp. 18 – 24.

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