A LIGHTWEIGHT APPROACH FOR SEMI-SUPERVISED SOUND EVENT DETECTION WITH UNSUPERVISED DATA AUGMENTATION

Heinrich Dinkel, Xinyu Cai, Zhiyong Yan, Yongqing Wang, Junbo Zhang, Yujun Wang

Xiaomi Corporation, Beijing, China
{dinkelheinrich,caixinyu,yanzhiyong, wangyongqing3,zhangjunbo1,wangyujun}@xiaomi.com

ABSTRACT

This paper describes our submission to the DCASE 2021 challenge. Different from most other approaches, our work focuses on training a lightweight and well-performing model which can be used in real-world applications. Compared to the baseline, our model only contains 600k parameters, resulting in a size of 2.7 Mb on disk, making it viable for applications on low-resource devices such as mobile phones. As a novelty, our approach uses unsupervised data augmentation (UDA) as the primary consistency criterion, which we show can achieve competitive performance to the more common mean teacher paradigm. Our submitted results on the validation set result in a single model peak performance of 36.91 PSDS-1 and 57.17 PSDS2, outperforming the baseline by an absolute of 2.7 and 5.0 points respectively. The best submitted ensemble system using a 5-way fusion achieves a PSDS-1 of 38.23 and PSDS-2 of 62.29 on the validation dataset. Our system ranks 7th in the official DCASE2021 Task4 challenge ranking and is the best performing model without post-processing while also having the least amount of parameters (3.4 M) by a large margin. Post-challenge evaluation reveals that by applying simple median post-processing, our approach achieves comparable performance to the 5th place.

Index Terms— Semi-supervised learning, Convolutional recurrent neural networks, Weakly supervised learning, unsupervised domain adaptation.

1. INTRODUCTION

This work focuses on modeling audio signals for sound event detection (SED). The main objective within SED is to categorize (i.e., tag) an event, with its respective on- and offsets. A core difficulty in this task is that multiple sound events can simultaneously occur during a time window.

One possible method to train a SED model is by using fully supervised labels, where on- and offsets for each event of interest are provided. However, obtaining fully supervised labels via manual labeling is expensive and thus might be a hindrance for SED systems at scale. To the best of our knowledge, there currently only exists a single large-scale manual labeled dataset, being Audioset [1], which provides full annotation for around 200 hours of data.

This paper focuses on semi-supervised sound event detection, where the provided training data is largely incomplete. Specifically, the DCASE2021 Task4 challenge focuses on low-cost sound event detection, where only a small fraction of data (4 hours) is manually weakly annotated. All other available data sources are either generated or do not contain labels.

Currently, SED can be used for a variety of applications, query-based sound retrieval [2, 3], smart cities, and homes [4, 5], voice activity detection [6, 7] as well as an important component of audio captioning [8, 9]. Most current approaches within SED utilize neural networks, in particular convolutional neural networks [10] (CNN), convolutional recurrent neural networks [11] (CRNN) and other models such as transformers and conformers [12, 13].

CNN models excel at audio tagging [14] and scale with data, yet falling behind CRNNs and transformer approaches in onset and offset estimations [15].

1.1. Problem statement

In the following, assume that x is an input (either raw-waveform or some spectrogram) and $\hat{y}$ is a predicted label.

Weakly supervised SED models commonly have two outputs: A clip-level prediction head $C(x) \mapsto \hat{y} \in \{0,1\}^E$ and a frame-level output $F(x) \mapsto \hat{y}_t \in \{0,1\}^E, t = 1, \dots, T$ for a frame at time t with E events. Both of these heads are directly connected via an aggregation function: $C(\cdot) = \text{agg}(F(\cdot))$, which summarizes the frame-level predictions to a single clip-level response. When training in strictly weakly supervised fashion, only the clip-level prediction head C can be learned, while F needs to be inferred by the model.

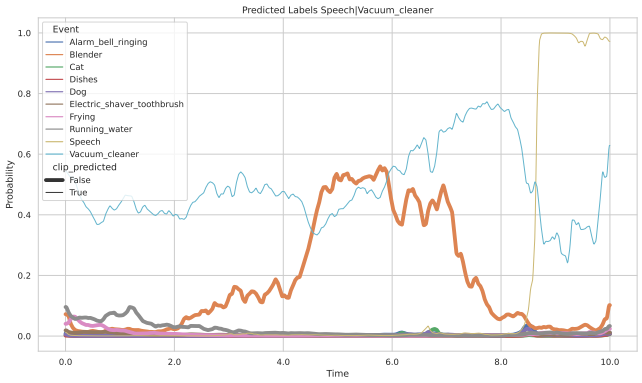

Figure 1: Inconsistent predictions between the two output heads in weakly-supervised SED are tackled in this work. The clip-level prediction $\hat{y}$ estimates the presence of "Speech" and "Vaccum cleaner", but the frame-level output $\hat{y}_t$ additionally predicts the presence of "Blender" (in bold) and some other noisy event outputs.

One of the key problems regarding training of weakly supervised SED models is that both heads can predict contradictory results since only errors in C are back-propagated, while F cannot be directly controlled. For example, the frame head F might predict

the presence of a sound event e during some time frame, due to factors such as noise or general similarity of a sound event to another (e.g., Blender, Vacuum Cleaner), while the clip head C predicts that e is not present. We show an example of this behavior in Figure 1, which is a prediction done by our baseline model.

Since in this challenge, the only human-annotated training data is provided on clip-level, meaning that the clip head C should provide reliable outputs, additional predictions from F can be considered as an inconsistency between both heads. In order to mitigate the inconsistency problem, we propose a simple learnable clip-smoothing algorithm.

2. PROPOSED APPROACH

2.1. Learnable clip-smoothing

We propose learnable clip-smoothing to combat the problem of inconsistent predictions the C and F heads. This technique is identical to clip-thresholding for weakly supervised SED [11], but since the DCASE2021 Task4 dataset provides strong frame-level labels, the clip-smoothing threshold can now be jointly optimized with the weak labels.

In particular, clip-smoothing is computed as in Equation (1), where $\hat{y}(t)^{\dagger}$ is the clip-smoothed output of our model for event e and $\hat{y}_t(e)$ is the model's frame head output (F):

$$\hat{y}_t^{\dagger}(e) = \hat{y}_t(e) * \hat{y}(e). \tag{1}$$

This approach should reduce false alarms, since the clip-level output will squash the frame-level probabilities for any non-occurring event. Subsequently, by using learnable clip-smoothing, the head F will only output events which also have a large probability score (≈ 1) in C.

2.2. Unsupervised data augmentation for consistency training

Many techniques exist to utilize unlabeled data to improve model performance. Mean Teacher (MT) [16] is a popular technique used in recent DCASE challenges [17].

In this work we propose the use of unsupervised domain (UDA) [18] for consistency training in SED. The advantages of UDA are:

1. In the vision domain, UDA has been seen to outperform other unsupervised methods such as MT [18].
2. Since only a single model is trained, performance evaluation is simpler compared to MT, where two models need to be evaluated.
3. We believe that the main contributing factor of MT is that it enables the usage of unlabeled data to improve performance. Our work shows that unsupervised data augmentation is equally effective in boosting performance.

The idea of UDA is to compute a consistency loss for unlabeled data between an augmented and a non-augmented (or differently augmented) sample. To the best of our knowledge, UDA has not been previously used in SED.

$$\begin{aligned} x^{\dagger} &= \text{Aug}(x), \\ \mathcal{M}(x) &\mapsto (\hat{y}, \hat{y}_t), \\ \mathcal{M}(x^{\dagger}) &\mapsto (\hat{y}^{\dagger}, \hat{y}_t^{\dagger}), \\ \mathcal{L}_{\text{UDA}}(x) &= \mathcal{L}_{\text{consistency}}(\hat{y}^{\dagger}, \hat{y}) + \mathcal{L}_{\text{consistency}}(\hat{y}_t^{\dagger}, \hat{y}_t). \end{aligned} \tag{2}$$

The UDA consistency training scheme is defined as in Equation (2). Here, a sample x is fed through a trainable neural network $\mathcal{M}$ where clip ($\hat{y}$) and frame-level ($\hat{y}_t$) predictions are obtained. The consistency between these predictions ($\hat{y}, \hat{y}_t$) and the predictions obtained by augmenting the input sample x denoted as $x^{\dagger}$ and predict ($\hat{y}^{\dagger}, \hat{y}_t^{\dagger}$) is the training objective. Note that in our work, we use UDA for both model heads, whereas it would be possible to use UDA for only weak or strong labels respectively. Also, it is worth mentioning that gradients are not computed during $\mathcal{M}(x)$.

3. EXPERIMENTAL SETUP

Log Mel-spectrogram (LMS) features are chosen as the default front-end feature for the task. Each 64-filter LMS is extracted from a 25 ms window with a stride of 10 ms, resulting in an approximately 1001×64 dimensional input tensor. If segments are shorter than 10 seconds, we zero-pad the input to the longest sample within a batch. During inference, we use a batch size of 1, such that padding has no effect on the final evaluation.

All experiments start with a learning rate of 0.001 and are run for at most 200 epochs, with a linear warmup duration of 20 batches using the Adam optimizer. The learning rate is halved every 1000 batches. Batch sizes are set to be 32 for weak and synthetic data and 64 for unlabeled data. The available weak training data is split into a 90% training and a 10% cross-validation portion. Cross-validation is done on the 10% held-out weak subset with the additional synthetic validation data. The training objective is the sum of the weak F1 and the intersection-F1 score, whereas training is stopped if the model did not improve for 15 epochs. Pytorch [19] was used as the neural network back-bone.[1].

3.1. Dataset

The dataset used in this work is the DCASE2021 dataset, which focuses on sound event detection in domestic environments.

The DCASE 2021 dataset is split into a development (used for training) and an evaluation section. The development set is further split into training and validation sections. The training section contrains three datasets $\mathcal{D}_{weak}, \mathcal{D}_{syn}, \mathcal{D}_{un}$, as seen in Equation (3).

$$\begin{aligned} \mathcal{D}_{weak} &= \{(x_1, y_2), (x_2, y_2), \ldots, (x_N, y_N)\}, \\ \mathcal{D}_{syn} &= \{(x_1, y_2), (x_2, y_2), \ldots, (x_M, y_M)\}, \\ \mathcal{D}_{un} &= \{x_1, \ldots, x_P\}. \end{aligned} \tag{3}$$

Note that the labels for $\mathcal{D}_{weak}$ are provided on clip-level, i.e., $y_j \in \{0,1\}^E, j \leq N$, while labels for $\mathcal{D}_{syn}$ are provided at frame-level, i.e., $y_k \in \{0,1\}^{ET}, k \leq M$ for each timestep in T. The unlabeled dataset contains only samples with target events also seen in the weak training data.

3.2. Model

Our model named CDur is a lightweight (in terms of parameters) 5-layer CRNN directly taken from the previous work in [11].

$$\hat{y} = \frac{\sum_t \hat{y}_t^2}{\sum_t \hat{y}_t} \tag{4}$$

[1]The source is available at `https://github.com/bibiaaaa/SmallRice_DCASE2021Challenge`

CDur subsamples the time-dimension by a factor of 4 and uses linear-softmax [20] as its aggregation method defined in Equation (4). The frame-level output is upsampled by a non-learnable transformation. For more information about the model, please refer to [11]. One of the benefits of the proposed model is its size, it only contains around 600k parameters, making it a lightweight alternative to the larger baseline model (1.1M parameters). Note however, that CDur requires slightly more FLOPs (3.36 GFlops) compared to the baseline (2.97 GFlops).

Three losses are used, one for each respective training data subset. Note that we experimented with additional losses such as asymmetric focal loss (AFL) [21], but did not observe gains in performance.

$$\mathcal{L}_{\text{sup}} = \text{BCE}(\hat{y}, y), \{y, \hat{y}\} \in \mathcal{D}_{weak}, \quad (5)$$

$$\mathcal{L}_{\text{syn}} = \text{BCE}(\hat{y}_t, y_t), \{y_t, \hat{y}_t\} \in \mathcal{D}_{syn}, \quad (6)$$

$$\mathcal{L}_{\text{unsup}} = \mathcal{L}_{\text{UDA}}(x) = \text{BCE}(\hat{y}^{\dagger}, \hat{y}) + \text{BCE}(\hat{y}_t, \hat{y}_t), x \in \mathcal{D}_{un}. \quad (7)$$

The model is optimized using the sums of all introduced losses seen in Equation (8).

$$\mathcal{L}_{\text{tot}} = \mathcal{L}_{\text{sup}} + \mathcal{L}_{\text{syn}} + \mathcal{L}_{\text{unsup}} \quad (8)$$

As the default in our work use UDA for both C and F heads. Augmentation in regards to UDA is applied on raw-wave level, where the torchaudio[2] and torch-audiomentations[3] packages are used. Specifically, we apply random *Gain* (in range -20, 10 db), *Polarityinversion* (with both probability 50%), and time masking (zeroing a sequence of at most 2 seconds, similar to SpecAug) to an input sample.

4. RESULTS

We report our results in terms of Event-F1 (E-F1) [22], Intersection-F1 (I-F1), and the two main challenge metrics denoted as PSDS-1 and PSDS-2 [23]. Additionally, we provide the d-prime d' score, which represents our model's capability to detect the presence of an event on clip-level and takes values in range $d' \geq 0$, where higher values are better.

Note that for *all* results, *no* post-processing is used and the Event-F1 score is calculated from the thresholded $\hat{y}_t > 0.5$ frame-predictions.

Data	d'	E-F1	I-F1	**PSDS-1**	**PSDS-2**
Weak	2.28	22.71	49.06	15.17	33.47
+ Syn	2.23	30.39	49.63	19.01	28.12
++ Unlabel	2.47	32.11	52.14	26.87	42.19

Table 1: Baseline results using CDur training with amounts of training data. All results are an average over 5 individual runs on the development dataset. Highlighted scores are the main challenge evaluation metrics. Higher is better.

The baseline experiments using the proposed CDur model can be seen in Table 1. The additional data synthetic data seems to decrease d', which likely stems from the mismatch between the synthetic and real data. With the addition of the unlabeled data, however, d' largely enhances, since the model now has access to larger amounts of real-world samples. This enhancement is then reflected on the PSDS-1 and PSDS-2 scores since the clip-smoothing technique's filtering capability is now enhanced.

Data	d'	E-F1	I-F1	**PSDS-1**	**PSDS-2**
Weak	2.27	22.99	49.14	19.98	46.57
+ Syn	2.21	35.31	54.84	29.85	47.34
++ Unlabel	2.50	37.21	57.12	34.41	54.90

Table 2: Development dataset results using the proposed clip-smoothing with CDur. All results are an average over 5 individual runs. Highlighted scores are the main challenge evaluation metrics. Higher is better.

Our results with the proposed clip-smoothing technique can be observed in Table 2. Comparing to our baseline, clip-smoothing leads to a large improvement for all metrics, leading to a comparable performance in terms of PSDS-1 and -2 against the strong baseline.

4.1. Data Augmentation

Two augmentation methods, namely SpecAug [24] and Mixup are used to enhance performance. The results can be seen in Table 3. Adding SpecAug to our model training decreases all metrics except PSDS-2, while the addition of SpecAug + Mixup shows improvements for both PSDS-1 and PSDS-2 scores. In the following, every experiment denoted as Aug uses SpecAug and Mixup as default.

Aug	d'	E-F1	I-F1	**PSDS-1**	**PSDS-2**
Base	2.50	37.21	57.12	34.41	54.90
+ SpecAug	2.64	35.68	57.06	32.60	56.26
++ Mixup	2.60	35.76	56.01	34.59	57.11

Table 3: Results with additional data augmentation in form of SpecAug and Mixup on the development dataset. All results are an average over 5 individual runs. Highlighted scores are the main challenge evaluation metrics.

4.2. Ensemble and submissions

The ensemble submissions seen in Table 4 named S1, S2 and S3 are frame-level averaged over the respective single models, which are:

- Aug, which uses clip-smoothing and additional specaug + mixup during training (see Table 3).
- *Heavy* uses much stronger augmentations during UDA than the default ones. Time Masking with a maximal length of 5s as well as a 70 % probability to apply volume gain in the range of -20 to 20 dB.
- *MSE* uses the mean square error criterion for UDA training instead of the default BCE.
- *WeakShift* Uses an additional augmentation via shifting of the time domain (with rollover) during UDA training. Note that the training criterion becomes $\mathcal{L}_{\text{UDA}} = \text{BCE}(\hat{y}^{\dagger}, \hat{y})$.
- *Sub-8* subsamples the time dimension by a factor of 8, leading to an output resolution of 80ms instead of 40ms.

[2] https://github.com/pytorch/pytorch

[3] https://github.com/asteroid-team/torch-audiomentations

Model	d'	E-F1	I-F1	**PSDS-1**	**PSDS-2**
Baseline	-	40.10	76.60	34.20	52.70
Aug (A)	2.66	36.80	58.94	33.63	57.43
Heavy (B)	2.56	39.02	58.09	35.21	58.00
MSE (C)	2.46	35.08	56.69	34.24	55.07
WeakShift (D)	2.50	39.29	59.02	36.91	57.17
Sub-8 (E)	2.66	36.05	57.17	33.00	59.38
S1 (A+B+C)	2.70	40.89	59.13	37.25	61.99
S2 (S1 + D)	2.70	40.90	59.61	**38.23**	62.29
S3 (S2 + E)	**2.75**	**41.06**	59.71	38.13	**62.98**

Table 4: Performance for the best single model results on the development dataset and the submitted ensemble models. Best results are highlighted in bold. Ensembles are generated by averaging the frame-level outputs of each respective model.

Compared to the baseline, our model falls behind in terms of Intersection-F1 and Event-F1, which is likely due to our neglect of post-processing methods largely affecting those metrics. However, in terms of PSDS, our model largely outperforms the baseline approach by an absolute of at least 3 and 9 points, respectively. Our submissions to the challenge include the ensemble systems S1, S2 and S3 as well as our best performing single model (D).

4.3. Challenge results

After the challenge ended, the results of all teams participated were published. Within the challenge, our method scored the 7th overall place in terms of the averaged PSDS-1 and PSDS-2 metrics as well as the average of both metrics (PSDS-Avg). A comparison of our method against other challenge participants can be seen in Table 5. Notably, our approach is the best performing approach in the challenge without requiring post-processing.

Model	**PSDS-1**	**PSDS-2**	**PSDS-Avg**	Post
Baseline	31.5	54.7	43.1	Median
1st	**45.2**	**74.6**	**59.9**	
2nd	44.2	67.4	55.8	
3rd	39.9	71.5	55.7	
3rd	41.9	68.6	55.2	Median
4th	41.6	63.7	52.6	
5th	41.3	58.6	49.9	
6th	37.0	62.6	49.8	
S1	36.1	58.4	47.2	
S2	37.3	58.5	47.9	
S3	37.0	59.6	48.3	-
S4 (Single)	33.9	50.4	42.1	
Ours (best)	**37.3**	**59.6**	**48.4**	-

Table 5: Performance of our models in comparison to other participants in the challenge on the official evaluation dataset. Best models with postprocessing (Post) using median filtering and without are displayed in bold.

4.4. Post-challenge post-processing

Since all other participants opt to use median filtering, we also provide our results on the development set in comparison to theirs using an adaptive window size for each event. The window sizes are estimated from the validation dataset, where one-third of each event's average duration is used as the window sizes.

Model	#Param (M)	**PSDS-1**	**PSDS-2**	**Score**	Single?
1st	14.3	**45.2**	**74.6**	**1.40**	N
2nd	20.2	44.2	67.4	1.32	Y
3rd	79.2	33.9	71.5	1.29	N
3rd	50.0	41.9	68.6	1.29	N
4th	119.8	41.6	63.7	1.24	N
S3	**3.4**	38.2	65.4	1.20	Y
S2	**2.7**	37.9	64.3	1.19	Y
5th	8.5	41.3	58.6	1.19	Y
S1	**2.0**	36.1	64.3	1.16	Y
6th	6.7	37.0	62.6	1.16	Y

Table 6: Post-challenge performance of our models in comparison to other participants using median post-processing on the evaluation set. "Single" refers to whether the results stem from a single submission or two different submissions. "Score" represents the challenge ranking score, where 1.0 is the challenge baseline. If multiple models were used, the reported parameter count represent the sum of each individual model's parameters.

As we can see from the results in Table 6, our model compares favorably against other participants in terms of parameter count to performance ratio[4]. Further, the submission S3 achieves a noticeable boost of 2 and 5 points in terms of PSDS-1 and PSDS-2 scores respectively on the evaluation dataset when using median filtering. Within the top-performing submissions, our proposed method is the most lightweight by a large margin as seen in Table 6, achieving comparable performance to the 4th place, while using only a fraction (2 %) of its parameters. Finally, our method ranks overall second if we only compare scores obtained by a single submission i.e., a model which performs well in terms of both PSDS-1 and PSDS-2 scores.

5. CONCLUSION

This paper proposes our submission to the DCASE2021 Task4 challenge. The approach uses clip-smoothing in combination with a small parameter model to outperform the provided baseline in terms of PSDS-1 and PSDS-2 scores. Our best single model achieves a PSDS-1 of 36.91 and 33.9 and a PSDS-2 of 57.17 and 50.4 on the validation and evaluation datasets, respectively. Moreover, our 4-model ensemble approach achieves a PSDS-1 of 38.23 and a PSDS-2 of 62.29, significantly outperforming the challenge baseline by an absolute of 4.03 and 9.6 points respectively. In terms of the official evaluation, our method scored seventh place, while being the only top-ranking method not using post-processing. When utilizing common adaptive median post-processing our approach achieves comparable performance to the 5th place, while having the fewest parameters amongst all top-ranked methods.

6. REFERENCES

[1] S. Hershey, D. P. W. Ellis, E. Fonseca, A. Jansen, C. Liu, R. Channing Moore, and M. Plakal, "The Benefit of

[4]We thank Romain Serizel for re-evaluating the post-processed results.

Temporally-Strong Labels in Audio Event Classification," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Institute of Electrical and Electronics Engineers (IEEE), may 2021, pp. 366–370.

[2] F. Font, G. Roma, and X. Serra, *Sound Sharing and Retrieval*. Springer International Publishing, 2018, pp. 279–301.

[3] A.-M. Oncescu, A. S. Koepke, J. F. Henriques, Z. Akata, and S. Albanie, "Audio Retrieval with Natural Language Queries," in *Proc. Interspeech 2021*, 2021, pp. 2411–2415.

[4] J. P. Bello, C. Mydlarz, and J. Salamon, *Sound Analysis in Smart Cities*. Cham: Springer International Publishing, 2018, pp. 373–397.

[5] S. Krstulović, *Audio Event Recognition in the Smart Home*. Cham: Springer International Publishing, 2018, pp. 335–371.

[6] Y. Chen, H. Dinkel, M. Wu, and K. Yu, "Voice activity detection in the wild via weakly supervised sound event detection," *Proc. Interspeech 2020*, pp. 3665–3669, 2020.

[7] H. Dinkel, S. Wang, X. Xu, M. Wu, and K. Yu, "Voice Activity Detection in the Wild: A Data-Driven Approach Using Teacher-Student Training," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1542–1555, 2021.

[8] X. Xu, H. Dinkel, M. Wu, Z. Xie, and K. Yu, "Investigating local and global information for automated audio captioning with transfer learning," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 905–909.

[9] M. Wu, H. Dinkel, and K. Yu, "Audio caption: Listen and tell," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 830–834.

[10] L. Lin, X. Wang, H. Liu, and Y. Qian, "Specialized Decision Surface and Disentangled Feature for Weakly-Supervised Polyphonic Sound Event Detection," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 28, pp. 1466–1478, may 2020.

[11] H. Dinkel, M. Wu, and K. Yu, "Towards duration robust weakly supervised sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 887–900, 2021.

[12] Q. Kong, Y. Xu, W. Wang, and M. D. Plumbley, "Sound Event Detection of Weakly Labelled Data with CNN-Transformer and Automatic Threshold Optimization," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 28, pp. 2450–2460, 2020.

[13] K. Miyazaki, T. Komatsu, T. Hayashi, S. Watanabe, T. Toda, and K. Takeda, "Convolution-augmented transformer for semi-supervised sound event detection," DCASE2020 Challenge, Tech. Rep., June 2020.

[14] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-Scale Pretrained Audio Neural Networks for Audio Pattern Recognition," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 28, pp. 2880–2894, dec 2020.

[15] N. Turpault, R. Serizel, and E. Vincent, "Limitations of weak labels for embedding and tagging," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 131–135.

[16] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 1195–1204.

[17] J. Yan, Y. Song, L.-R. Dai, and I. McLoughlin, "Task-Aware Mean Teacher Method for Large Scale Weakly Labeled Semi-Supervised Sound Event Detection," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings 2020*. Institute of Electrical and Electronics Engineers (IEEE), apr 2020, pp. 326–330.

[18] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised Data Augmentation for Consistency Training," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2019, pp. 6256–6268.

[19] 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 *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8026–8037.

[20] Y. Wang, J. Li, and F. Metze, "A Comparison of Five Multiple Instance Learning Pooling Functions for Sound Event Detection with Weak Labeling," *ICASSP 2019 - ICASSP 2019 IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2019-May, pp. 31–35, oct 2019.

[21] K. Imoto, S. Mishima, Y. Arai, and R. Kondo, "Impact of sound duration and inactive frames on sound event detection performance," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 860–864.

[22] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences (Switzerland)*, vol. 6, no. 6, p. 162, may 2016.

[23] C. Bilen, G. Ferroni, F. Tuveri, J. Azcarreta, and S. Krstulovic, "A Framework for the Robust Evaluation of Sound Event Detection," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, may 2019, pp. 61–65.

[24] 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 *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2019-September. International Speech Communication Association, 2019, pp. 2613–2617.