A MUTUAL LEARNING FRAMEWORK FOR FEW-SHOT SOUND EVENT DETECTION

Dongchao Yang[1], *Helin Wang*[1], *Yuexian Zou*[1,*], *Zhongjie Ye*[1], *Wenwu Wang*[2]

[1]ADSPLAB, School of ECE, Peking University, Shenzhen, China
[2]Center for Vision, Speech and Signal Processing, University of Surrey, UK

ABSTRACT

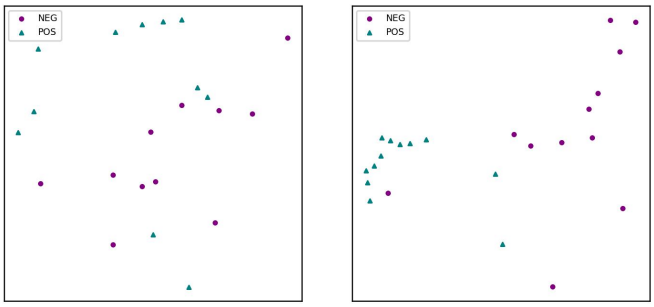
Although prototypical network (ProtoNet) has proved to be an effective method for few-shot sound event detection, two problems still exist. Firstly, the small-scaled support set is insufficient so that the class prototypes may not represent the class center accurately. Secondly, the feature extractor is task-agnostic (or class-agnostic): the feature extractor is trained with base-class data and directly applied to unseen-class data. To address these issues, we present a novel mutual learning framework with transductive learning, which aims at iteratively updating the class prototypes and feature extractor. More specifically, we propose to update class prototypes with transductive inference to make the class prototypes as close to the true class center as possible. To make the feature extractor to be task-specific, we propose to use the updated class prototypes to fine-tune the feature extractor. After that, a fine-tuned feature extractor further helps produce better class prototypes. Our method achieves the F-score of 38.4% on the DCASE 2021 Task 5 evaluation set, which won the first place in the few-shot bioacoustic event detection task of Detection and Classification of Acoustic Scenes and Events (DCASE) 2021 Challenge.

Index Terms— Few shot learning, transductive inference, sound event detection, mutual learning

1. INTRODUCTION

Deep learning-based sound event detection methods typically require large amounts of data for training or fine-tuning models for specific applications [1, 2, 3]. The development of deep learning models to detect unseen sound classes with only few labels is insufficient. Recently, studies [4, 5] have proposed to tackle this problem using few-shot learning (FSL), where a classifier needs to learn to recognize novel classes given only few samples of each class. In the FSL setting, a model is first trained on labeled data with base classes. Then, model generalization is evaluated on few-shot tasks, composed of unlabeled samples from novel classes unseen during training (query set), assuming only one or a few labeled samples (support set) are given per novel class. Prototypical network (ProtoNet) [6] has been proved as an effective method for few-shot sound event detection [7, 8]. In DCASE 2021 Challenge Task 5, the official baseline and several solutions [9, 10] submited to this challenge have also employed ProtoNet. However, there are still two factors that limit the performance of ProtoNet. Firstly, the class feature of the support set may be insufficient due to the presence of background noise and interference in audio data, so that the class prototypes learned from such support set may not represent the class center accurately. Figure 1 shows the learned representations (embeddings) extracted from ProtoNet, and we can see that the embeddings of each class are scattered, especially for the support set of 'buk4.wav', which contains more background noise than 'a1.wav'. As a result, the F-score of 'buk4.wav' is much lower than that of 'a1.wav'. Secondly, ProtoNet trains a feature extractor with the base-class data and applies the feature extractor to samples from unseen classes. This style of transfer learning is task-agnostic: the feature extractor is not learned to be optimally discriminative with respect to the unseen classes. It often performs worse than a task-specific feature extractor [11, 12].

Following the discussions and observations above, we



(a) Samples from the support set of 'buk4.wav' (b) Samples from the support set of 'a1.wav'

Fig. 1. The visualization of the embeddings of the support set from test audio file by t-SNE [13]. We choose two audios ('buk4.wav' and 'a1.wav'), each audio includes two classes (POS and NEG). The overall F-score values of 'buk4.wav' and 'a1.wav' are 45.5% and 63.7% respectively.

This paper was partially supported by the Shenzhen Science & Technology Fundamental Research Programs (No: JCYJ20180507182908274 & JSGG20191129105421211) and GXWD20201231165807007-20200814115301001.

* Corresponding Author: zouyx@pku.edu.cn

propose a mutual learning framework to continuously update the feature extractor and class prototypes. More specifically, we firstly train a feature extractor with base-class data and use the class prototypes to initialize a classifier. We then leverage the statistics of unlabelled audio to update the classifier with transductive inference [14, 15, 16]. In order to obtain a task-specific feature extractor, we further use the updated class prototypes as the supervised information to fine-tune the feature extractor. These processes can be repeated several times so that the feature extractor and classifier can be continuously updated. Our contributions can be summarized as follows: (1) To solve the problem that class prototypes cannot represent the true class centers accurately, we propose to update class prototypes with transductive learning. (2) To make the feature extractor to be task-specific, we propose a novel method to fine-tune the feature extractor. (3) Our mutual learning framework significantly improves the performance of few-shot bioacoustic event detection over the state-of-the-art methods.

2. PROPOSED METHOD

In this section, few-shot setting, transductive inference and the mutual learning framework will be introduced.

2.1. Few-shot setting

Assume we are given a training set, $X_{base} = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N_{base}}$, where $\boldsymbol{x}_i$ denotes the acoustic feature of example i and $\boldsymbol{y}_i$ denotes associated one-hot label. Let Y_{base} denote the label set of this base dataset. The few-shot learning assumes that we are given a test dataset: $X_{test} = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N_{test}}$, with a completely new label set Y_{test} such that $Y_{base} \cap Y_{test} = \emptyset$, and the test dataset only has a few labelled examples.

2.2. Transductive inference

Transductive inference (TI) is about reasoning from observed, specific (training) cases to specific (test) cases. In this paper, the core idea of TI is about leveraging the statistics of the unlabeled data. More specifically, we adapt the idea from [14], which maximizes the mutual information (MI) between the query features and their label predictions for a few-shot task at inference. It means that the model has seen these unlabeled data before making final prediction.

2.3. Mutual learning framework

The overview of the mutual learning framework is shown in Figure 2. In this section, we first introduce how to use class prototypes to build a classifier and update the classifier with transductive inference. After that, we discuss how to make use of the updated class prototypes to fine-tune the feature extractor. Lastly, we summarize the core idea of mutual learning framework.

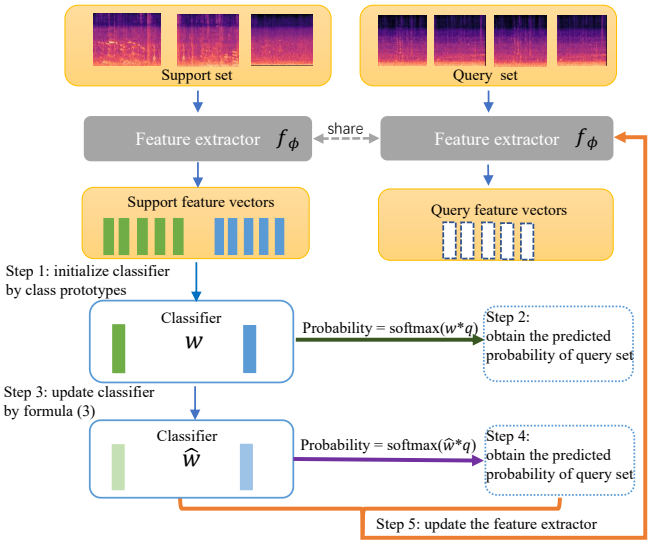


Fig. 2. The overview of mutual learning framework. Feature extractor is trained with base class data.

Building classifier For a given few-shot task, with a support set S and a query set Q, let X denote the random variables associated with the acoustic features within $S \cup Q$ and let $Y = \{1, 2, ..., K\}$ be the random variables associated with the labels. Let $f_\phi : X \to Z \subset R^d$ denote the encoder (*i.e.*, feature extractor) function of a deep neural network, where ϕ denotes the trainable parameters, and Z stands for the set of embedded features. The encoder is firstly trained from the base training set X_{base} using the standard cross-entropy loss. Next, for each specific few-shot task, we define a classifier, parametrized by a weight matrix $\boldsymbol{W} = [\boldsymbol{w}_1, ..., \boldsymbol{w}_K] \in R^{K \times d}$. The posterior distribution over labels given features is defined by $p_{ik} = \mathbb{P}(Y = k | X = \boldsymbol{x}_i; \boldsymbol{W}, \phi)$. The marginal distribution over query labels is defined by $\hat{p}_k = \mathbb{P}(Y_Q = k; \boldsymbol{W}, \phi)$. p_{ik} and $\hat{p}_k$ are calculated as formula (1).

$$p_{ik} = \frac{exp(\boldsymbol{w}_k \cdot \boldsymbol{z}_i)}{\sum_{c=1}^{K} exp(\boldsymbol{w}_c \cdot \boldsymbol{z}_i)}, \hat{p}_k = \frac{1}{Q} \sum_{i \in Q} p_{ik} \tag{1}$$

where $\boldsymbol{z}_i = \frac{f_\phi(\boldsymbol{x}_i)}{||f_\phi(\boldsymbol{x}_i)||_2}$ denotes L2-normalized embedded features. For each task, weights $\boldsymbol{W}$ are initialized by the class prototypes of the support set, as follows

$$\boldsymbol{w}_k = \frac{1}{|S_k|} \sum_{\boldsymbol{x}_i \in S_k} f_\phi(\boldsymbol{x}_i) \tag{2}$$

In this paper, we only need to judge whether the audio frame is a positive sample, so we set $K = 2$.

Updating classifier To update the weight matrix $\boldsymbol{W}$, for each single few-shot task, we propose a loss function with two complementary terms: (1) a standard cross-entropy loss on the support set; (2) a mutual-information loss, which includes a conditional entropy loss and a marginal entropy loss.

$$L_w = \lambda_{CE} \cdot CE - \hat{I}(X_Q; Y_Q) \tag{3}$$

$$CE = -\frac{1}{|S|}\sum_{i\in S}\sum_{i=1}^{K} y_{ik} log(p_{ik}) \tag{4}$$

$$\hat{I}(X_Q; Y_Q) = -\sum_{k=1}^{K}\hat{p}_k log\hat{p}_k + \frac{1}{|Q|}\sum_{i\in Q}\sum_{k=1}^{K} p_{ik} log(p_{ik}) \tag{5}$$

where CE denotes the cross entropy loss function, y_{ik} denotes the true label of the sample in the support set, p_{ik} denotes the prediction result. In our experiments, λ_{CE} is set as 0.1. $\hat{I}(X_Q; Y_Q)$ denotes the mutual information between the query samples and their latent labels. It is a combination of two terms, the first term is the empirical label-marginal entropy, denoted as $\hat{H}(Y_Q)$, while the second term is an empirical estimate of the conditional entropy of labels given the query acoustic features, denoted as $\hat{H}(Y_Q|X_Q)$. $\hat{H}(Y_Q|X_Q)$ aims at minimizing the uncertainty of the posteriors at unlabelled query samples, thereby encouraging the model to output confident predictions. This entropy loss is widely used in the context of semi-supervised learning (SSL) [17, 18], as it models effectively the cluster assumption: the classifier's boundaries should not occur at dense regions of the unlabelled features. The label-marginal entropy regularizer $\hat{H}(Y_Q)$ encourages the marginal distribution of labels to be uniform.

Note that we only update the weight matrix $\boldsymbol{W}$ in this step, while the feature extractor is fixed. Our experimental results also show that simultaneously updating feature extractor f_ϕ and weight matrix $\boldsymbol{W}$ does not offer better performance.

Updating feature extractor Previous works [11, 12] have shown that a task-specific feature extractor works better than a task-agnostic one, so we expect our feature extractor to be task-specific. To achieve this, we propose a novel method to update feature extractor, which uses the updated class prototypes as supervision information to fine-tune the feature extractor. In addition, we plan to make use of the predicted results for unlabelled data. Figure 2 shows the updating process of our method. When we finish step 3 and 4, we make use of $\hat{\boldsymbol{W}}$ and predicted results of high confidence to fine-tune the feature extractor f_ϕ. The loss function has two terms as formula (6) shows, including a cross-entropy (CE) loss according to pseudo label and a contrastive loss.

$$L_f = \lambda_1 * CE + \lambda_2 * L_c \tag{6}$$

where λ_1 and λ_2 are hyper-parameters. In our experiments, $\lambda_1 = \lambda_2 = 0.5$. The contrastive loss L_c is defined as follows.

$$L_c = -log(\frac{exp(sim(\hat{\boldsymbol{w}}[1], \bar{\boldsymbol{z}}_{pos}))}{\sum_{k=1}^{N} exp(sim(\hat{\boldsymbol{w}}[1], z_{neg}^k))}) \tag{7}$$

where $\hat{\boldsymbol{w}}[1]$ denotes the first row vector of $\hat{\boldsymbol{W}}$, and it represents the prototype of positive class in our experiments. $\bar{\boldsymbol{z}}_{pos}$ denotes the mean of the learned representation of the positive samples on the support set, and z_{neg} denotes the learned representation of negative samples. N denotes the number of negative samples. In our experiments, sim stands for cosine similarity. We do not use $\hat{\boldsymbol{w}}[2]$ for the reason that the negative sample is randomly chosen. This loss function also can be viewed as knowledge distillation [19].

Mutual learning According to previous discussion, we can make use of transductive inference to improve the classifier, and we can also improve feature extractor by the updated classifier and pseudo label. After we get a better feature extractor, we can continue running the previous process to update the classifier. It means that feature extractor and classifier can learn from each other, so we name it as mutual learning.

3. EXPERIMENT

3.1. Experimental setups

Dataset The dataset is from DCASE2021 task 5 [20], including development and evaluation sets. The development set is pre-splitted into training and validation sets. The training set contains about 14 hours of audio, and the validation set contains 5 hours of audio. The evaluation set consists of 31 audio files acquired from different bioacoustic sources.

Metrics For all the experiments, we use the event-based F-measure [21] as the evaluation metric, which is one of the most commonly used metrics for sound event detection.

Preprocessing All the raw audios are down-sampled to 22.05 kHz and applied a Short Time Fourier Transform (STFT) with a window size of 1024 samples, followed by a Mel-scaled filter bank on perceptually weighted spectrograms. This results in 128 Mel frequency bins and around 86 frames per second. The input frames are normalized to zero-mean and unit variance according to the training set.

Training We use the same backbone as the baseline [6], which only includes 4 convolutional layers. The only difference is that we do not use meta-learning training strategy. Instead, we directly train feature extractor by the cross entropy loss. Specifically, we use a dense layer after the feature extractor, and then add a softmax layer to get classification probability. The Adam optimizer is applied for a total of 15 epochs, with an initial learning rate of 1×10^{-3}.

Updating classifier The test audio only gives the first five positive annotations, and negative samples are randomly sampled from unlabelled parts. In order to update the classifier $\boldsymbol{W}$, the Adam optimizer is used for a range of 5-30 epochs, with an initial learning rate of 1×10^{-5}. We choose different training epoch for different test audio, for the reason that training epochs will affect the prediction results. The prediction results at the last epoch are used as our final results.

Updating feature extractor We build a new dense layer after the feature extractor, which only need to do binary classification task. The Adam optimizer is used for a total of 5 epochs, with an initial learning rate of 1×10^{-4} for feature extractor, and 1×10^{-3} for the new dense layer.

Table 1. F-score comparison of different methods on DCASE 2021 task5 Development and Evaluation dataset.

Method	Dev-set	Eval-set
Baseline [6]	41.48	20.1
Anderson *et al.* [9]	26.2	35.0
Tang *et al.* [10]	51.4	38.3
TI (ours)	51.21	33.2
TI-ML (ours)	**55.26**	**38.4**

Table 2. Ablation study on the effect of each term in formula (3). CE: Cross entropy loss, I: Mutual information loss.

Method	Loss	Precision	Recall	F-score
TI	*None*	16.89	60.1	26.38
	I	57.8	43.9	49.96
	CE	55.17	46.07	50.21
	I+CE	57.11	46.39	**51.21**
TI-ML	*None*	15.5	50.5	23.76
	I	69.6	43.6	53.67
	CE	52.68	49.1	50.83
	I+CE	65.54	47.76	**55.26**

3.2. Experimental results

Table 1 shows the experimental results. Our method achieves 38.4 % F-score on evaluation set, which significantly outperforms the baseline [6]. TI denotes we only use transductive learning to update classifier (class prototypes). TI-ML denotes we use mutual learning framework to update classifier and feature extractor. TI-ML performs better than TI, which shows the effectiveness of our mutual learning framework. Anderson *et al.* [9] also applied ProtoNet, and compared with baseline [6], their model utilized both per-channel energy normalisation (PCEN) on the front end and three data augmentation methods. In contrast, our approach does not need to employ these strategies and gets a better result. In addition, Tang *et al.* [10] got a very close result with us on the evaluation dataset, but they used a 12-layer ResNet pre-trained on AudioSet [22] as the backbone.

3.3. Ablation study

In this part, we discuss the influence of transductive learning and mutual learning. The experiments were carried out on the development set.

Influence of each term on formula (3) We now assess the impact of each term in formula (3). The results are reported in Table 2. We observe that integrating the two terms in our loss consistently outperforms any other configuration. *None* indicates that we do not update classifier $\boldsymbol{W}$, otherwise we directly use the class prototypes to initialize the classifier, and then use it to predict results. We first analyze their effect on transductive inference (TI). If we do not update the classifier, the F-score is 26.38%. The performance is lower than baseline [6] for the reason that we do not use the meta-learning training strategy. On the contrary, when we use either cross entropy loss or mutual information loss to update the classifier, the performance will be significantly improved. We can find that only using cross-entropy loss brings improvement. It proves that insufficient support set leads to the problem that class prototypes cannot represent the true class center, because class prototypes cannot even properly classify the support set. Secondly, we analyze their effect on the mutual learning (TI-ML). If we use the classifier which is not updated to fine-tune feature extractor, we will get the worst results. If we only use the cross entropy loss to update the classifier, it brings a small improvement. Compared with the cross entropy loss, the mutual information loss has more advantages on mutual learning. The best results can be obtained when both two terms are used.

Influence of each term on formula (6) To fine-tune the feature extractor, we devise a loss function as shown in (6) which includes a cross the entropy loss and a contrastive loss. If we only use the cross entropy loss, the result is 51.64%. If the contrastive loss is used alone, the result is 51.06%. The best result (55.26%) can be obtained when both are used, which shows that these two loss terms are both important.

Influence of iterations Table 3 reports the impact of iterations for mutual learning. Here, the iteration equals to 0 means we do not update the feature extractor. Experimental results indicate updating it only once offers the best F-score. We find that the performance tends to decline when the number of iterations increase, one of the reasons is that the number of negative samples is far more than that of positive samples in this dataset, which makes the model learn more information about negative samples, so the result shows higher False Positives (FP) rate.

Table 3. Ablation study on the effect of iterations on mutual learning.

Method	Iterations	Precision	Recall	F-score
TI-ML	0	57.12	46.39	51.21
	1	65.54	47.76	**55.26**
	2	72.50	43.38	54.28
	3	69.53	43.68	53.66

4. CONCLUSIONS

In this paper, we proposed a mutual learning framework with transductive inference to continuously improve the ability of feature extractor and classifier. Our method won the first place in the DCASE 2021 Challenge Task 5, with a F-score of 38.4%. In the future, we will further improve the performance of our systems. The source code is released.[1]

[1] https://github.com/yangdongchao/DCASE2021Task5

5. REFERENCES

[1] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2016, pp. 6440–6444.

[2] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. Plumbley, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[3] 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.

[4] J. Pons, J. Serrà, and X. Serra, "Training neural audio classifiers with few data," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 16–20.

[5] S. Chou, K. Cheng, J. Roger Jang, and Y. Yang, "Learning to match transient sound events using attentional similarity for few-shot sound recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 26–30.

[6] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 4077–4087.

[7] Y. Wang, J. Salamon, N. Bryan, and J. P. Bello, "Few-shot sound event detection," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 81–85.

[8] B. Shi, M. Sun, K. C. Puvvada, C. Kao, S. Matsoukas, and C. Wang, "Few-shot acoustic event detection via meta learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 76–80.

[9] R. Bielecki, "Few-shot bioacoustic event detection with prototypical networks, knowledge distillation and attention transfer loss," *DCASE Challenge*, 2021.

[10] T. Tang, Y. Liang, and Y. Long, "Two improved architectures based on prototype network for few-shot bioacoustic event detection," *DCASE Challenge*, 2021.

[11] H. Ye, H. Hu, D. Zhan, and F. Sha, "Few-shot learning via embedding adaptation with set-to-set functions," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8808–8817.

[12] P. Bateni, R. Goyal, V. Masrani, F. Wood, and L. Sigal, "Improved few-shot visual classification," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 14493–14502.

[13] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[14] M. Boudiaf, I. Ziko, J. Rony, J. Dolz, P. Piantanida, and I. B. Ayed, "Information maximization for few-shot learning," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020.

[15] M. Boudiaf, H. Kervadec, Z. I. Masud, P. Piantanida, I. B. Ayed, and J. Dolz, "Few-shot segmentation without meta-learning: A good transductive inference is all you need?," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 13979–13988.

[16] Y. Liu, J. Lee, M. Park, S. Kim, E. Yang, S. Hwang, and Y. Yang, "Learning to propagate labels: Transductive propagation network for few-shot learning," *International Conference on Learning Representations (ICLR)*, 2019.

[17] G. Yves and B. Yoshua, "Semi-supervised learning by entropy minimization," in *In Advances in Neural Information Processing Systems (NeurIPS)*, 2004, pp. 281–296.

[18] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, and Oliver, "Mixmatch: A holistic approach to semi-supervised learning," in *Neural Information Processing Systems (NeurIPS)*, 2019.

[19] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[20] V. Morfi, D. Stowell, V. Lostanlen, A. Strandburg-Peshkin, L. Gill, H. Pamula, D. Benvent, I. Nolasco, S. Singh, S. Sridhar, M. Duteil, and A. Farnsworth, "DCASE 2021 Task 5: Few-shot Bioacoustic Event Detection Development Set," Feb. 2021.

[21] A. Mesaros, Toni Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, pp. 162, 2016.

[22] J. Gemmeke, D. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 776–780.