FEW-SHOT BIOACOUSTIC EVENT DETECTION: A GOOD TRANSDUCTIVE INFERENCE IS ALL YOU NEED

Technical Report

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

[1] Peking University, School of ECE, shenzhen, china
[2]Peng Cheng Laboratory, Shenzhen, China
dongchao98@stu.pku.edu.cn, wanghl15@pku.edu.cn, zhongjieye@stu.pku.edu.cn, zouyx@pku.edu.cn

ABSTRACT

In this technical report, we describe our few-shot bioacoustic event detection methods submitted to Detection and Classification of Acoustic Scenes and Events Challenge 2021 Task 5. We analyze the reason why Prototypical networks cannot perform well, and propose to use transductive inference for few shot learning. Our method maximizes the mutual information between the query features and their label predictions for a given few-shot task, in conjunction with a supervision loss based on the support set. Furthermore, we propose a mutual learning framework, which makes feature extractor and classifier to help each other. Experimental results indicate our transductive inference method get better performance than baseline, and F1 score is about 50.8% on evaluation set. Furthermore, our mutual learning framework brings about 5% improvement over the transductive inference method. We will release our code on `https://github.com/yangdongchao/DCASE2021Task5`.

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

1. INTRODUCTION

Few-shot learning [1, 2, 3] is a highly promising paradigm for sound event detection. It is also an extremely good fit to the needs of users in bioacoustics, in which increasingly large acoustic datasets commonly need to be labelled for events of an identified category (e.g. species or call-type), even though this category might not be known in other datasets or have any yet-known label.

Few-shot learning describes tasks in which an algorithm must make predictions given only a few instances of each class, contrary to standard supervised learning paradigm. The main objective is to find reliable algorithms that are capable of dealing with data sparsity, class imbalance and noisy or busy environments. Few-shot learning is usually studied using N-way-K-shot classification, where N denotes the number of classes and K denotes the number of examples for each class.

Few-shot learning tasks have been increasingly studied in literature and often rely on meta-learning approaches including MAML (Model-Agnostic Meta-Learning) [4], Prototypical network [5], Relation network, and so on. Most such works are done in computer vision [6, 3] or natural language recognition [7] while very little work has been done in audio-related tasks. In [8], authors compare different few shot methods on acoustic event detection, which shows Prototypical network gets better performance. In the challenge of DCASE2021 task5, the official baseline [9] also chooses Prototypical network.

Nowadays, transductive inference methods have get SOTA performance in computer vision fields [10, 11]. Does it also work better on audio tasks? Inspired by Transductive Infomation Maximization (TIM) [10], we propose to use transductive inference to solve few-shot problem in audio tasks.

In this report, We introduce a transductive inference, which leverages the statistics of the unlabeled audio of a task by optimizing a new loss containing three 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; (3) a global KL-divergence regularizer based on the proportion of positive sample and negative sample. We use these terms to update our soft-classifier, which is parametrized by weight matrix, and the weight matrix is initialized by the mean of support feature vectors (category prototype). The reason for updating weight matrix is that support samples cannot represent the category. Because the support set is too small and most of the samples are incomplete, the mean of these feature may be far from the true category center. This is why Prototypical network cannot perform well, and we carefully check these audio files, we find if the labeled part is difficult to identify by human, the Prototypical network will have bad performance on this audio, otherwise if these labeled part is clear, Prototypical networks can get good performance, because the mean of these feature can represent category center. Furthermore, we also propose a method to update feature extractor, because we want our feature extractor is task-dependent. But updating feature extractor is difficult, it may cause performance decrease. In this challenge, we propose use the updated classifier and pseudo label to update feature extractor. The reason for using the updated classifier is that we expect our feature extractor produce feature can directly use as the weight matrix of classifier, in other words, we want get a better classifier so that we never need to update it by another loss.

2. MOTIVATIONS

2.1. Prototypical networks

Most of the existing approaches within the FSL framework are based on the "learning to learn" paradigm or meta-learning, where the training set is viewed as a series of balanced tasks (or episodes), so as to simulate test-time scenario. Prototypical network [5] is one of successful examples using meta-learning. In this task, the baseline chooses Prototypical network. The main idea of Prototypical network is using few-shot prototypes as class center, and then we

judge unlabeled data belongs to which class according to their L_2 distance to class center. Few-shot prototypes are computed as the mean of embedded support examples for each class.

According to previous description, we can find that a good prototype representation is very important. After training, we can believe our network can extract class-wise information from data, but the problem is most of the samples are incomplete, such as incomplete samples, background interference or fuzzy details, so that some representative attribute characteristics are lost. This is why Prototypical networks cannot perform well on some tasks. In this challenge, evaluation set give 8 different audio files, we find that if the labeled part (support sets) has good quality, the results of Prototypical network is very good. So the key point is to find a good prototype representation for these incomplete support samples.

2.2. Motivation

As previous discussion, incomplete support sets data will lead to prototype cannot represent the category center. One way to solve this problem is finding supplementary information, which can help prototype representation as close to the true category center as possible. In this report, we propose to leverages the statistics of the unlabeled audio to update the prototype representation, this method also called as transductive inference.

Furthermore, we also note that Feature extractor is trained from base class data, in other words, the feature extractor is task-independent. We expect our feature extractor is task-dependent, but if we fine-tune feature extractor only according to the support data, it is easy to overfit. To solve this problem, we propose a mutual learning framework, which can help classifier and feature extractor to improve each other. Specifically, we use the updated classifier and pseudo label of query set to fine-tune feature extractor, and then we use the updated Feature extractor and transductive inference updating classifier again. This process can be iterated on and on, so we call it as mutual learning framework. The details can refer to Section 4.

3. TRANSDUCTIVE INFERENCE

3.1. Few-shot setting

Assume we are given a labeled training set, $X_{base} = \{\boldsymbol{x_i}, \boldsymbol{y_i}\}_{i=1}^{N_{base}}$, where $\boldsymbol{x}_i$ denotes raw features of sample i and $\boldsymbol{y_i}$ denotes associated one-hot encoded label. Let Y_{base} denotes the set of classes for this base dataset, and in our development set, we have 19 different classes. The few-shot scenario assumes that we are given a test dataset: $X_{test} = \{\boldsymbol{x_i}, \boldsymbol{y_i}\}_{i=1}^{N_{test}}$, with a completely new set of classes Y_{test} such that $Y_{base} \cap Y_{test} = \emptyset$, from which we create randomly sampled few-shot tasks, each with a few labeled examples. Specifically, each K-way N-shot task involves sampling N_S labeled examples from each of K different classes, also chosen at random. Let S denote the set of these labeled examples, referred to as the support set with size $|S| = N_S \cdot K$. Furthermore, each task has a query set denoted by Q composed of $|Q| = N_Q \cdot K$ unlabeled (unseen) examples from each of the K classes. With models trained on the base set, few-shot techniques use the labeled support sets to adapt to the tasks at hand, and are evaluated based on their performances on the unlabeled query sets.

3.2. Proposed formulation

In this part, we introduce some basic notation and definitions before presenting the overall loss and the optimization strategies.

For a given few-shot task, with a support set S and a query set Q, let X denotes the random variable associated with the raw features within $S \cup Q$ and let $Y = \{1, 2....K\}$ be the random variable associated with the labels. Let $f_\phi : X \to Y \subset R^d$ denotes 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 first trained from the base training set X_{base} using the standard cross-entropy loss, without any meta training or specific sampling schemes. Then, for each specific few-shot task, we propose to minimize mutual-information loss and KL loss defined over the query samples.

Formally, we define a soft-classifier, parametrized by weight matrix $\boldsymbol{W} \in R^{K \times d}$, whose posterior distribution over labels given features, $p_{ik} := P(Y = k | X = \boldsymbol{x_i}; \boldsymbol{W}, \phi)$, and marginal distribution over query labels, $\hat{p}_k = P(Y_Q = k; \boldsymbol{W}, \phi)$, are given by formula (1).

$$p_{ik} \propto exp(\boldsymbol{w_k} \cdot \boldsymbol{z_i}), \hat{p}_k = \frac{1}{Q} \sum_{i \in Q} p_{ik} \tag{1}$$

Where $\boldsymbol{W} := [\boldsymbol{w}_1, ..., \boldsymbol{w}_K]$ denotes classifier weights, $\boldsymbol{z}_i = \frac{f_\phi(\boldsymbol{x}_i)}{||f_\phi(\boldsymbol{x}_i)||_2}$ the L2-normalized embedded features. For each task, weights $\boldsymbol{W}$ are initialized by the class prototypes of support set.

$$\boldsymbol{w}_k = \frac{\sum_{i \in S} y_{ik} \boldsymbol{z}_i}{\sum_{i \in S} y_{ik}} \tag{2}$$

3.3. Loss function

Now, for each single few-shot task, we propose to update weight matrix $\boldsymbol{W}$ by following loss function:

$$L_w = \lambda \cdot CE - \hat{I}(X_Q; Y_Q) + \lambda_{KL} * D_{KL} \tag{3}$$

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

Where CE denotes cross entropy loss function, y_{ik} denotes the true label of sample in support set, p_{ik} denotes the predict result. $\hat{I}(X_Q; Y_Q)$ denotes the mutual information between the query samples and their latent labels, its definition as formula (5) shows. D_{KL} denotes the KL loss between predicted proportion of positive samples and true proportion of positive samples, its definition as formula (6) shows.

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

$$D_{KL} = \hat{p}_Q \cdot log(\frac{\hat{p}_Q}{\boldsymbol{\pi}}) \tag{6}$$

$$\hat{p}_Q = \begin{cases} \frac{1}{|Q|} \sum_{i \in Q} \mathbb{I}(p_i > 0.5), & \text{Positive sample proportion} \\ 1 - \frac{1}{|Q|} \sum_{i \in Q} \mathbb{I}(p_i > 0.5), & \text{Negative sample proportion} \end{cases} \tag{7}$$

Where p_i denotes the probability of i-th sample contains positive events. $\boldsymbol{\pi} \in [0, 1]^2$ denotes the proportion of positive sample and negative sample.

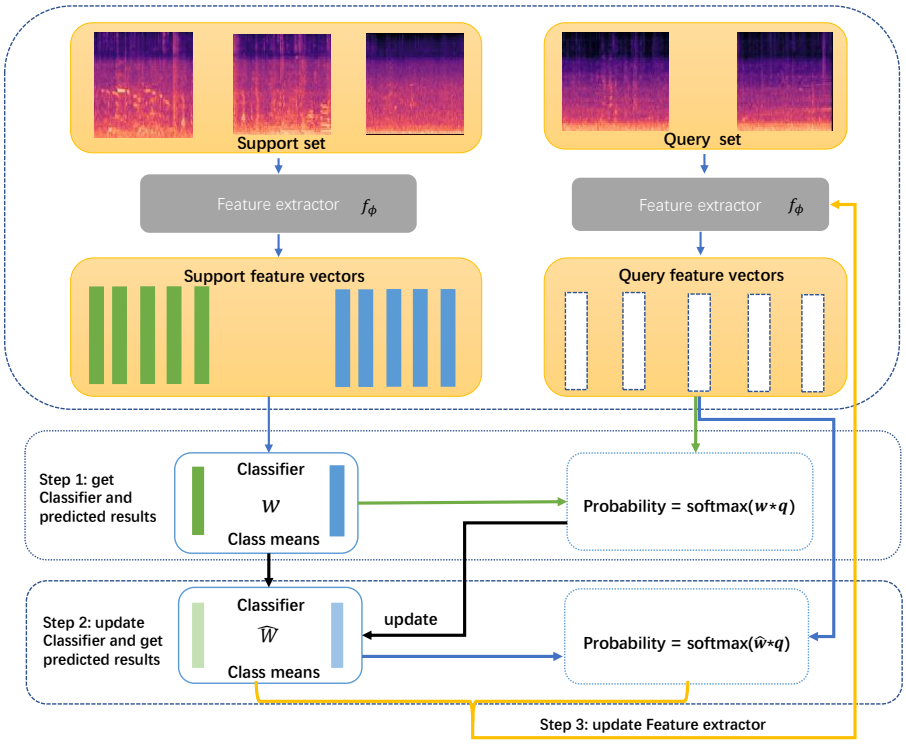

Figure 1: The framework of our proposed method.

The mutual information loss is combined by two terms, the first term is an empirical (Monte-Carlo) estimate of the conditional entropy of labels given the query raw features, denoted $\hat{H}(Y_Q|X_Q)$, while the second term is the empirical label-marginal entropy, $\hat{H}(Y_Q)$. $\hat{H}(Y_Q|X_Q)$ aims at minimizing the uncertainty of the posteriors at unlabeled query samples, thereby encouraging the model to output confident predictions. This entropy loss is widely used in the context of semi-supervised learning (SSL) [12, 13], as it models effectively the cluster assumption: The classifier's boundaries should not occur at dense regions of the unlabeled features. The label-marginal entropy regularizer $\hat{H}(Y_Q)$ encourages the marginal distribution of labels to be uniform, thereby avoiding degenerate solutions obtained when solely minimizing conditional entropy. The details please refer to TIM [10].

The idea of minimizing D_{KL} loss is from Few-Shot Segmentation [11], authors propose to predict the portion of background and foreground. In our report, we propose to predict the proportion of positive sample and negative sample. But we never known the true proportion, so we use previous predict results $\hat{p}_Q^{(t-1)}$ as true proportion. The definition of D_{KL} also can be viewed as formula (8).

$$D_{KL} = \hat{p}_Q^{(t)} \cdot log(\frac{\hat{p}_Q^{(t)}}{\hat{p}_Q^{(t-1)}}) \tag{8}$$

Note we only update the weight matrix $\boldsymbol{W}$, and the feature extractor is fixed, and our experimental results also show that simultaneously update feature extractor f_ϕ and weight matrix $\boldsymbol{W}$ cannot get better performance. Figure 1 shows the diagram of our proposed method.

4. MUTUAL LEARNING FRAMEWORK

In this part, we will systematic introduction of our proposed method.

According to previous description, we can get a better prototype representation by using transductive inference. But our feature extractor is trained from base class, which means it is task-independent. We think a task-dependent extractor is more helpful for our few-shot setting, but if we only use support set to fine-tune our feature extractor, it is easy to overfit. So it is important to find a new way to fine-tune feature extractor.

4.1. Update Feature Extractor

Our aim is to find a good prototype by extracted feature from support set, and now we already have a better prototype after updating classifier. We expect our feature extractor can produce better prototype directly. In other words, we expect our feature extractor can produce prototype as similar as the updated prototype. Furthermore, we can also make use of previous predicted result, which have high predict confidence. Figures 1 shows the process of our methods, when we finish step 2, and then we make use of $\hat{\boldsymbol{W}}$ and high confidence predicted label to fine-tune Feature extractor f_ϕ. The loss function as formula (9) shows, which has tow terms. The first term is cross entropy loss according to pseudo label, and the second term is contrast loss.

$$L_f = \lambda_1 * CE + \lambda_2 * L_c \tag{9}$$

Where λ_1 and λ_2 are hyper-parameters, in our experiment $\lambda_1 = \lambda_2 = 0.5$. CE denotes the cross entropy loss, the contrast loss L_c

as formula (7) shows.

$$L_c = -log(\frac{exp(sim(\hat{\boldsymbol{w}}[0], \bar{\boldsymbol{z}}_{pos}))}{\sum_{k=1}^{N} exp(sim(\hat{\boldsymbol{w}}[0], z_{neg}))}) \quad (10)$$

Where $\hat{\boldsymbol{w}}[0]$ denotes the first vector of $\hat{\boldsymbol{W}}$, it represents the prototype of positive class in our experiments. $\bar{\boldsymbol{z}}_{pos}$ denotes the mean of the support feature, we want the feature extractor can produce prototype as close as $\hat{\boldsymbol{w}}[0]$. z_{neg} denotes the negative feature from pseudo label. We do not use $\hat{\boldsymbol{w}}[1]$ for the reason that negative prototype is difficult to fixed.

4.2. Mutual Learning

According to previous section, we can make use of transductive inference improve classifier, we can also improve feature extractor by updated classifier and pseudo label. After we get better feature extractor, we can continue run the previous process to get new classifier, and so on. It means feature extractor and classifier can help each other, so we name it as mutual learning.

5. EXPERIMENT

5.1. Dataset and metrics

Dataset The dataset is from DCASE2021 task5 evaluation set. **Metrics** For all the experiments, we use the event-based F-measure as the evaluation metric, which is one of the most commonly used metrics for sound event detection.

5.2. Setups

For training, we follow the same settings as the baseline [9]. Our feature extractor as same as baseline backbone. Specifically, the input i down-sampled to 22.05kHz and applied a Short Time Fourier Transform (STFT) with a window size of 1024, 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. The Adam optimizer [14] is used for a total of 15 epochs, with an initial learning rate of 0.0001. The learning rate decays linearly from epoch 10. The difference is that we never use meta-learning training strategy, otherwise we directly train feature extractor by cross entropy loss. Furthermore, we do not use any ensemble approaches.

For inference, to update the parameter of classifier, the Adam optimizer [14] is used for a range of 1-50 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 predicted results. The last epoch predicted results are used as our final predicted results.

For mutual learning, we build a new liner classifier for feature extractor. The Adam optimizer [14] is used for a range of 5 epochs, with an initial learning rate of 1×10^{-4} for feature extractor, the learning rate of new liner classifier is 50 times feature extractor. We will select those predictions with high confidence as pseudo-labels.

5.3. Experimental results

Table 1 shows the experimental results, which indicate our proposed method is very useful. TIM denotes we only use transductive learning to update classifier. TIM-ML denotes we use mutual learning framework to update classifier and feature extractor. Considering the cost of time, we only iterate one time, and our experimental results also show iterating one time almost keep same performance as iterating two times.

Table 1: F-score comparison of different methods on DCASE 2021 task5 evaluation set.

Method	precision	recall	F-score
Baseline	58.27	32.2	41.48
TIM(ours)	55.53	46.89	50.85
TIM-ML(ours)	**65.54**	**47.76**	**55.26**

6. CONCLUSIONS

In this report, we analyze the reason for Prototypical network cannot perform well is that incomplete support set data will lead to prototype cannot represent the category center. So we propose to supplement the prototype with transductive inference. Experimental results on the evaluation set validate the advantages of transductive inference. Furthermore, we also propose a mutual learning framework, which can further improve model performance.

7. REFERENCES

[1] E. G. Miller, N. E. Matsakis, and P. A. Viola, "Learning from one example through shared densities on transforms," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR*, vol. 1. IEEE, 2000, pp. 464–471.

[2] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum, "One shot learning of simple visual concepts," in *Proceedings of The Annual Meeting of The Cognitive Science Society*, vol. 33, no. 33, 2011.

[3] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML Deep Learning Workshop*, vol. 2. Lille, 2015.

[4] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1126–1135.

[5] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems 30*, 2017, pp. 4077–4087.

[6] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," 2016.

[7] M. Yu, X. Guo, J. Yi, S. Chang, S. Potdar, Y. Cheng, G. Tesauro, H. Wang, and B. Zhou, "Diverse few-shot text classification with multiple metrics," *arXiv preprint arXiv:1805.07513*, 2018.

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

[9] https://github.com/c4dm/dcase-few-shot-bioacoustic/tree/main/baselines.

[10] M. Boudiaf, Z. I. Masud, J. Rony, J. Dolz, P. Piantanida, and I. B. Ayed, "Transductive information maximization for few-shot learning," *arXiv preprint arXiv:2008.11297*, 2020.

[11] 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?" *arXiv preprint arXiv:2012.06166*, 2020.

[12] Y. Grandvalet, Y. Bengio, *et al.*, "Semi-supervised learning by entropy minimization." in *CAP*, 2005, pp. 281–296.

[13] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," *arXiv preprint arXiv:1905.02249*, 2019.

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