FEW-SHOT BIOACOUSTIC EVENT DETECTION USING A PROTOTYPICAL NETWORK ENSEMBLE WITH ADAPTIVE EMBEDDING FUNCTIONS

Technical Report

John Martinsson[1,2*], *Martin Willbo*[1], *Aleksis Pirinen*[1], *Olof Mogren*[1], *Maria Sandsten*[2]

[1]Computer Science, RISE Research Institutes of Sweden, Sweden
{john.martinsson, martin.willbo,aleksis.pirinen,olof.mogren }@ri.se
[2] Centre for Mathematical Sciences, Lund University, Sweden
{maria.sandsten}@matstat.lu.se

ABSTRACT

In this report we present our method for the DCASE 2022 challenge on few-shot bioacoustic event detection. We use an ensemble of prototypical neural networks with adaptive embedding functions and show that both ensemble and adaptive embedding functions can be used to improve results from an average F-score of 41.3% to an average F-score of 60.0% on the validation dataset.

***Index Terms*—** Machine listening, bioacoustics, few-shot learning, ensemble

1. INTRODUCTION

In few-shot bioacoustic event detection, the task is to predict the start time and the end time of certain bioacoustic events in a set of sound recordings from natural environments. The few-shot test set contains recordings for which only the first five examples of the bioacoustic event class of interest has been annotated, hence few-shot, and the goal is to detect the remaining events of this class in the rest of the recording. In figure 1 we show the setup of the task for one of the task recordings with predictions from our method. We are also given a base training set with annotated bioacoustic events of classes which are disjoint from the classes in the few-shot test set. During development, the few-shot test set is emulated using a supplied few-shot validation set where all events have been annotated as well, but where only the first five are used to infer the remaining events.

The need for solutions to this problem is motivated by the increasing amounts of audio data which are being recorded through acoustic monitoring devices, and where automated analysis is necessary to go through all of the collected data. Annotation efficient methods which can learn from very little annotated data is promising way forward.

The key contributions in this work are as follows:

- we train an embedding function to solve a multi-class sound event detection task since two different sound events can occur (partially) at the same time,
- we adapt the embedding function to the bioacoustic events we want to detect at inference time using the few-shot examples,
- we use ensemble predictions from multiple models trained on different time-frequency transforms to reduce false-positives, and
- we perform a comparison of three different time-frequency transformations.

*Thanks to the Swedish foundation for strategic research for funding.

2. METHOD

In this section we present our method, which is an ensemble of prototypical neural networks [1] with adaptive embedding functions. We describe how each embedding function is trained, how these are adapted using the few-shot examples, and how they are used to produce an ensemble prediction at test time.

2.1. Training embedding function

The base training data consists of sound recordings with annotations for 47 known event classes and one "unknown" event class. We are given the start and end times $\mathcal{A} = \{(s_i^k, e_i^k)\}_{i=1}^N$ of these classes, where (s_i^k, e_i^k) denotes the start and end time of sound event class k for annotation i. We model the 47 known sound event classes and the "unknown" sound event class in the same way, which yields a total of $K = 48$ classes. The goal is to learn an embedding function from this base training data. There is overlap in the annotations, i.e. two different sound events can occur (partially) at the same time, and we therefore treat this as a multi-class problem.

We assume a fixed length audio segment $x \in \mathbb{R}^T$ that consists of T consecutive audio samples is fed to the embedding function $f_\theta^T : \mathbb{R}^T \to \mathbb{R}^M$ (see section 2.4 for further details), where $M \ll T$. We split the audio recordings into audio segments $x_i \in \mathbb{R}^T$ by sliding a window of size T with a hop size of $T/2$ over each recording. For each audio segment x_i, a target vector $y_i \in \{0,1\}^{K \times n}$ is derived. If $n = T$ it means that the target contains one label per audio sample. Choosing $n < T$ means that the temporal resolution for the target is reduced. This results in a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ which defines the sound event detection task used to train the embedding function.

A prediction of the target classes for a given audio segment x_i is derived by $\hat{y}_i = h_\phi(f_\theta^T(x_i))$, where $h_\phi(\cdot)$ is a linear layer followed by an element-wise sigmoid activation function, and $f_\theta^T(\cdot)$ is a convolutional neural network where the first layer is a (non-learnable) time-frequency transform.

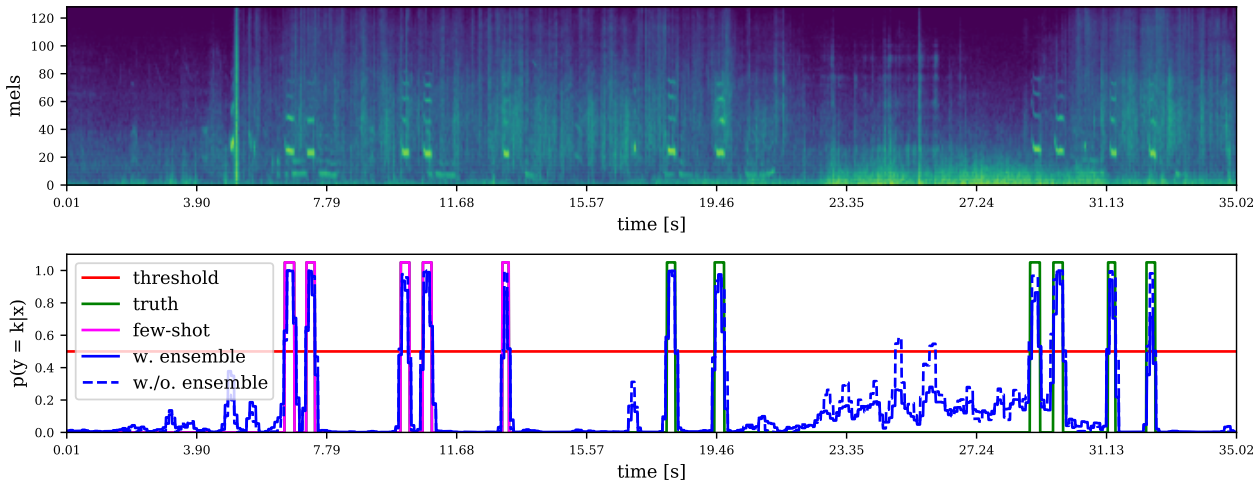

Figure 1: A log Mel spectrogram of part of a sound recording (top) and examples of predictions (bottom) from an ensemble prototypical network (solid blue line) and a prototypical network (dashed blue line) as well as the given few-shot examples (purple line) and remaining ground truth events (green line). The decision threshold τ of 0.5 (red line).

The loss function is the mean element-wise binary cross-entropy between the target y_i and the prediction $\hat{y}_i$, where the mean is taken over the class dimension K and the temporal dimension n.

For a fixed T, we train a set of C different embedding functions, all together parametrized as $\Theta = \{\theta_1, \ldots, \theta_C\}$, by varying the randomly initialized weights of the neural network, the training and validation split of the base training data, and the time-frequency transform in the first layer of the embedding function.

2.2. Prototypical network at test time

At test time we are given a sound recording and the $M = 5$ first examples of the class of interest. We denote these $A_p = \{(s_i, e_i)\}_{i=1}^{M}$ and call them the *positive* sound events. We assume that the gaps between these annotations are background noise and let $A_n = \{(e_i, s_{i+1})\}_{i=1}^{M-1}$ denote the start and end time of the $M-1$ first *negative* sound events. This is not necessarily true since an annotator may miss events, but we assume the likelihood of this to be low.

Let $l_i = e_i - s_i$ be the length of annotation i. If $l_i < T$ we "expand" the annotation with the $(T - l_i)/2$ preceding and subsequent audio samples to get an audio segment of length T and then we split this into segments of length T by sliding a window of size T over the signal with a hop size of $T/16$. Let S_p denote the set of positive audio segments derived from these annotated start and end times, and let S_n denote the set of negative audio segments. We use the embedding function f_θ^T and define the prototypes as

$$c_k = \frac{1}{|S_k|} \sum_{x \in S_k} f_\theta^T(x) \tag{1}$$

and derive a pseudo-probability of audio segment x belonging to sound class k from the prototypical network by

$$p_\theta(y = k|x) = \frac{\exp(-d(f_\theta^T(x), c_k))}{\sum_{k'} \exp(-d(f_\theta^T(x), c_{k'}))}, \tag{2}$$

where $k \in \{n, p\}$ and $d(z_i, z_j)$ denotes the Euclidean distance between z_i and z_j.

The query set S_q is derived by sliding a window of size T over the signal with a hop size of $T/2$. The reason for setting the hop size relative to T is that this means that we do equally many predictions for each audio sample in the validation recordings.

2.3. Our contributions

We now present the two main contributions of this paper: i) adapting the embedding function, and ii) using an ensemble of predictions.

Adapting the embedding function. We use the annotated positive events $A_p = \{(s_i, e_i)\}_{i=1}^{M}$ and compute the set of event lengths $L = \{e_i - s_i\}_{i=1}^{M}$. We choose $T \in \{T_1, 2^1 T_1, 2^2 T_1, 2^3 T_1\}$ such that $\sqrt{(T - l_{\text{min}}/2)^2}$ is minimized, where l_{min} is the shortest event length in L.

We choose $T_1 = 2048$ which is 0.09 seconds at a sampling rate of 22050 Hz so that we can plausibly detect the shortest events in the few-shot validation set. We limit the amount of memory needed during training and inference by only doubling up to three times. We have not extensively evaluated the effect of these choices and adding embedding functions trained on even shorter and longer segments may be beneficial, but of course comes at a computational cost during training.

Ensemble. Let $\Theta = \{\theta_i\}_{i=1}^{C}$ denote the set of parameters of C different adaptive prototypical network models. Then we define

$$p_\Theta(y = k|x) = \frac{1}{C} \sum_{\theta \in \Theta} p_\theta(y = k|x) \tag{3}$$

as in [2], which can be viewed as a uniformly-weighted mixture of experts. We say that x belongs to the positive event class if $p_\Theta(y = p|x) > \tau$ and the negative otherwise. This is done for every $x \in S_q$. Finally, if the query is classified as positive event then the start and end time associated with that query is used as the predicted positive event timings.

2.4. Details of the embedding function

The embedding function consists of a time-frequency transform followed by a convolutional neural network, both of which are briefly described below.

Time-frequency transform. The first layer of the embedding function is a time-frequency transform. Let $E(t, f)$ denote a Mel spectrogram with 128 Mel bins derived from an audio segment x. Then

$$\mathrm{S}(t, f) = 10 \log_{10} \frac{E(t, f)}{E_{\max}}, \tag{4}$$

and

$$\mathrm{PCEN}(t, f) = \left(\frac{E(t, f)}{(\epsilon + (E^t * \phi_T)(t, f))^\alpha} + \delta \right)^r - \delta^r. \tag{5}$$

We either use $\mathrm{S}(t, f)$ as the first layer embedding function, or we use one of two different parameter configurations for per-channel energy normalization (PCEN) [3], one of which is developed for speech audio and one of which is developed for bioacoustics as suggested in [4].

Convolutional neural network. The convolutional neural network used is an adapted version of the 10-layer residual neural network [5] implementation used in the baseline model for the challenge. Specifically, we i) add the classification head $h_\phi(\cdot)$ so that we can model the defined multi-class task, ii) use the same number of filters in every convolutional layer, and iii) reduce the max pooling along the time-dimension when audio segments are too short.

2.5. Post-processing

Since we get one prediction $x \in S_q$ of size T for each query audio segment, this limits how long predictions we can make with the model. To solve this, we simply merge all overlapping predicted positive events into one detected event with a single start and end time.

A predicted positive event will only be considered to be a match with a true positive event during evaluation if they have an intersection-over-union (IoU) of at least 0.3. We therefore remove predictions which are shorter than $0.3 * l_{\text{avg}}$ or longer than $(1/0.3) * l_{\text{avg}}$, where l_{avg} is the average event length of the given five annotations. Since predictions of these lengths can on average not be matched with true events as the evaluation is defined.

3. DATA

To highlight the types of variation that the model needs to handle, we have used the few-shot examples to compute the mean event length, the mean gap length, and the density of these sound events – see table 1. The mean event length is defined as the mean length of the five annotated events; the mean gap length is defined as the mean length of the *unannotated* gaps between the five annotated events; and the density is the sum of the time of the five annotated events divided by the total time spanned by the start of the first annotated event and the end of the last annotated event.

The validation set consists of three different subsets: HB, ME, and PB. We present the statistics for each subset in table 1. The HB subset contains long events with low noise, where the events are longer than the gaps between them, and as a result has a very high event density. The ME subset contains short events with low noise, where the events are shorter than the gaps between them, and as a result has a low event density. The PB subset contains very short events with very high noise, where the events are much shorter than the gaps between them, and has a result a very low event density.

Subset	Mean event length	Mean gap length	Mean density
HB	11.25 ± 3.11	6.12 ± 5.39	0.73 ± 0.12
ME	0.22 ± 0.03	1.40 ± 0.04	0.17 ± 0.02
PB	0.12 ± 0.08	59.89 ± 55.55	0.01 ± 0.02

Table 1: Validation data statistics.

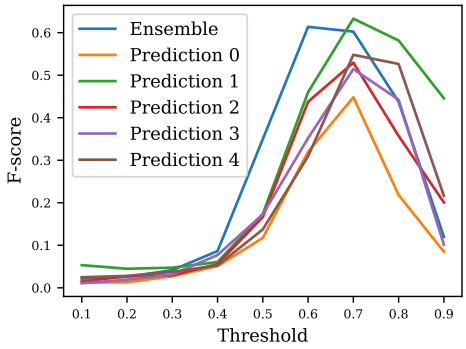

Figure 2: Comparing an ensemble of five predictions using embedding functions trained on PCEN (biodiversity) time-frequency transformations of the audio segments with each of these predictions itself.

4. EXPERIMENTS AND RESULTS

We have trained each embedding function using the Adam [6] optimizer with a learning rate of $1e-3$. The network is trained on a random split with 80% training data and validated on the remaining 20%. The training proceeds until we have observed no reduction in validation loss for the last 10 epochs and the model with the lowest validation loss is chosen as the final model. The temporal span of the targets have been fixed to $n = 16$, meaning that we have 16 targets for any given audio segment.

In figure 2 we compare the F-measure achieved on the few-shot validation set when using an ensemble of five predictions with using each of these predictions by themselves. The achieved F-measure by the ensemble is better than the best of these individual predictions for $0.4 \leq \tau \leq 0.6$, and outperforms or matches the mean of them for other τ. We also note that the optimal τ is around 0.7 for the single predictions, and moves to 0.6 for the ensemble.

In figure 3 we compare the F-measure achieved on the few-shot validation set for an ensemble over five predictions for each time-frequency transform with using an ensemble over both time-frequency transforms and the five predictions for each time-frequency transform (an ensemble over 15 predictions). We do not observe a significant increase in F-measure when comparing the ensemble to the ensembles using the PCEN (bioacoustic) time-frequency transforms, but it outperform the one using the PCEN (speech) and log Mel transform. The optimal threshold τ varies around 0.6 to 0.7 for the ensembles using a single transform, and is

at 0.6 for the ensemble.

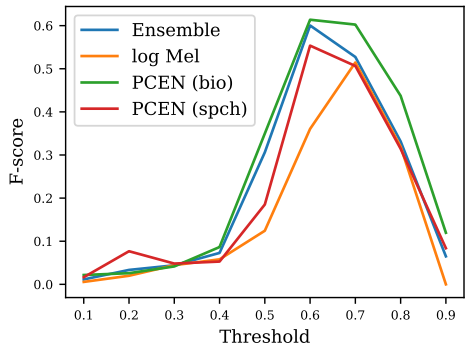

Figure 3: Comparing ensembles using embedding functions trained and tested on log Mel, PCEN (bioacoustics), or PCEN (speech), with an ensemble prediction of using all these time-freuency transforms.

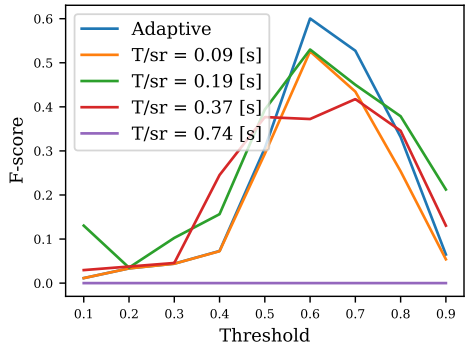

Figure 4: Comparing the adaptive embedding function with using each of the fixed size embedding functions respectively. sr denotes the sample rate, and T denotes the length of the audio segment.

In figure 4 we compare the F-measure achieved on the few-shot validation set when using the adaptive embedding functions in the ensemble with using any of the fixed $T \in \{T_1, 2^1T_1, 2^2T_1, 2^3T_1\}$. Adapting the embedding function increases performance from 53.0% (using best $T = 4096$) to 60.0% F-score for $\tau = 0.6$.

In table 2 we show an ablation where performance of a prototypical network using a embedding function (no ensemble) which has been trained on PCEN (speech) and a fixed segment length of 4096 is compared to using an adaptive embedding function, and then performing an ensemble prediction over the time-frequency transforms and random initialization of network weights and training/validation split (3×5). Adapting the embedding function increases the F-score on average with 8.3 percentage points, and adding the ensemble increases the F-score an additional 11.4 percentage points.

In table 3 we compare the results of the baselines with the F-score on the few-shot validation set for each of the systems for which we have submitted predictions on the test data for the challenge.

Method	Ensemble	Adaptive	F-score [%]
Ours	No	No	41.3 ± 3.8
Ours	No	Yes	49.6 ± 5.3
Ours	Yes	Yes	60.0

Table 2: An ablation of our system where we add adaptive embedding functions and ensemble.

Submission	τ	Ensemble + Adaptive	F-score [%] (valid)
Baseline (TM)	-	No	4.28
Baseline (PN)	0.5	No	29.59
Martinsson_1	0.6	True	60.0
Martinsson_2	0.5	True	30.6
Martinsson_3	0.6	False	44.6
Martinsson_4	0.5	False	13.3

Table 3: The validation scores for the baselines provided by the challenge organizers: template matching (TM) and prototypical networks (PN), and the validation score for the systems which have been submitted for test evaluation in the challenge.

5. DISCUSSION AND CONCLUSIONS

During development of this method we observed that random sampling of S_n, the set of negative examples, as in previous work does not work well for validation files with high event densities, which is why we chose to use the gaps between the first five annotated events instead.

We further observed that using one single fixed audio segment size T can be problematic. If T is much larger than the events we want to detect, the predictions will become too long to be counted as matches with the true events. Conversely, if T is much smaller than the events we want to detect, it may not cover the semantics which we want to detect. Therefore, choosing a T which matches the lengths of the events seem to be important. The adaptation of the embedding function could possibly be even stronger if based on both the event length statistic and the gap length statistic.

We observed that the optimal threshold was different for different validation files. We therefore try to calibrate the pseudo-probability in the predictions using an ensemble so that we get the best results by setting $\tau = 0.5$ as intended. We see from figure 2 and figure 3 that the optimal threshold τ moves from around $0.7 - 0.8$ to 0.6 which is what we want to achieve.

The ensemble improves performance by still correctly predicting most true positives, while no longer predicting as many false positives. This could intuitively be thought of as the ensemble being in agreement for true positive predictions, the average of which still yields a high pseudo-probability, while being in disagreement when predicting false positives, the average of which would be closer to 0.5. This effect can be seen in figure 1, where some of the false positives predicted when not using an ensemble (dashed blue line) are removed by using an ensemble of the predictions (solid blue line), leading to a reduction in false-positives.

In conclusion, we have shown that adapting the embedding function to the event lengths we want to detect can increase performance, and that false-positives can be reduced by an ensemble of predictions. We have also shown that out of the three time-frequency transforms we have studied, PCEN (bioacoustics) perform best for this bioacoustics task, followed by PCEN (speech) and then by log Mel.

6. REFERENCES

[1] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in Neural Information Processing Systems*, vol. 2017-Decem, pp. 4078–4088, 2017.

[2] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in Neural Information Processing Systems*, vol. 2017-December, pp. 6403–6414, 2017.

[3] Y. Wang, P. Getreuer, T. Hughes, R. F. Lyon, and R. A. Saurous, "Trainable frontend for robust and far-field keyword spotting," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, no. 1, pp. 5670–5674, 2017.

[4] V. Lostanlen, J. Salamon, M. Cartwright, B. Mcfee, A. Farnsworth, S. Kelling, and J. P. Bello, "Per-Channel Energy Normalization: Why and How," *IEEE SIGNAL PROCESSING LETTERS*, no. September, pp. 1–6, 2018. [Online]. Available: http://www.justinsalamon.com/uploads/4/3/9/4/4394963/lostanlen_pcen_spl2018.pdf

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Arxiv.Org*, vol. 7, no. 3, pp. 171–180, 2015. [Online]. Available: http://arxiv.org/pdf/1512.03385v1.pdf

[6] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," pp. 1–15, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980