WIDE RESNET MODELS FOR FEW-SHOT SOUND EVENT DETECTION

Technical Report

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

[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

ABSTRACT

In this technical report we describe our few-shot sound event detection (SED) systems used to generate predictions for the *DCASE 2022 task 5 challenge*. At the core of the SED systems is a wider variant of ResNet-18, i.e., each block throughout the depth of the network have more convolutional filters. In addition to this, for one of the submissions we include what we believe to be a novel approach to semi-supervised learning for prototypical networks. For both the fully supervised and semi-supervised methods we showcase the importance of calibrating the probability thresholds in the few-shot learning tasks, and provide a simple implementation of how to find these.

Index Terms— CNN, Prototypical networks, Semi-supervised, Probability calibration

1. INTRODUCTION

This technical report provides an outline of our four submissions. The submissions are named as follows:

- willbo_supervised_1
- willbo_supervised_2
- willbo_semi_1
- willbo_semi_2

All submissions are based on prototypical networks [1] and share network architecture for the embedding function, which is described in Section 3. Other similarities between the submissions are described in Section 4.

The submissions titled *_2 use an adaptive threshold method described in Section 6 while submissions titled *_1 do not. Submissions titled *_semi_* make use of a semi-supervised prototypical network method described in Section 5.

2. DATA

Only the provided development set for the DCASE 2022 Task 5 challenge was used in training and validating the systems used for submissions.

The audio files in the Development and Evaluation sets were pre-processed in the same fashion as for the deep learning (prototypical network) baseline, with the exception of the *librosa core PCEN* implementation parameters. These were set to:

- *gain* : 0.8
- *bias* : 10
- *power* : 0.25
- *time-constant* : 0.06
- *eps* : 1e-6

This was done for all three sets (training, validation, and evaluation).

3. BACKBONE

For the backbone, the embedding function in the prototypical network setup, a modified ResNet-18 [2] model was used. First, instead of having three channels in the input, one channel was used. This is because the spectrograms fed to the network only have one channel depth-wise. Second, a larger (4x) amount of filters per block was used, resulting in a wider network. The network has approximately 180M parameters.

4. SHARED METHODOLOGY OF SUBMISSIONS

For all submissions, the predictions for test audio segments were made with a negative prototype based on an average of the whole test audio file, instead of a random sampling as done in the deep learning baseline for the challenge. The positive prototype was made with all available segments resulting from pre-processing the audio from the five provided annotations. That is, for audio files with longer events resulting in more than five spectrograms, all spectrograms were averaged over to create the positive prototype.

All submissions use the same post-processing, which consists of filtering short events (as done in the deep learning baseline), and median filtering with a window size set to one third of the average event length based on the first five annotations.

5. SEMI-SUPERVISED PROTOTYPICAL NETWORKS

This approach relies on the assumption that two temporally neighbouring audio segments share the same semantics. This assumption is likely to hold for recordings mostly containing background noise such as rain, wind and other natural phenomena. It also holds mostly true for events of lengths longer than the segment length of 0.2s, especially when considering the substantial overlap between segments. Including unlabeled parts of the audio segments with this in mind could enable a learner to make use of more of the provided data. This can be done by including *unlabeled* support and query

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

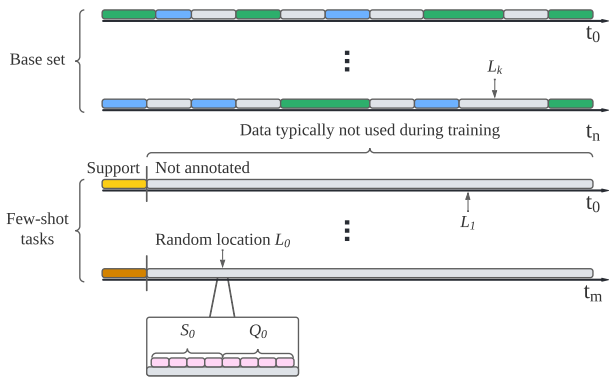

Figure 1: Overview of our training method for the submissions titled *_semi_*. Base set: Annotated segments of audio files corresponding to events (not background) colored in green and blue. Unlabeled segments (background) are colored in gray. Few-shot tasks: The support (annotated segments) in the beginning of the audio files are colored in yellow and orange. The rest of the audio files do not have annotations and are typically not included during training on the base set. In contrast, we propose to take advantage of the structure that exists also in this unlabeled data. To do this, locations (marked $L_0, L_1, \ldots, L_k$) in the audio files of both sets are first sampled uniformly at random. At these locations consecutive segments (pink) are then extracted to constitute *'unlabeled'* support and query sets, S_i and Q_i, from location L_i. These are finally included in an episode during training on the base set, just as if they were samples from an additional class.

sets in the episodic training scheme. See Figure 1 for a visualization of this. Note that such *unlabeled* support and query sets were sampled only from the base data set for this challenge, in contrast to what the figure shows.

For ecoacoustic recordings it may not be uncommon that two random audio segments from a recording environment are similar even though they are temporally separated – this is true for a recording that contains an hour of rain, for example. Including the suggested *unlabeled* support and query samples in the prototypical network loss, see equations (1) and (2), would lead to the prototypes based on these to be treated as different sound events and therefore pushed apart in the embedding space. We therefore introduce an alternative to the prototypical loss which is more suited for this scenario. This loss encourages distance between a labeled prototype and all other prototypes in the embedding space, while an unlabeled prototype is only encouraged to be distant from other labeled prototypes. See (3), where $\mathcal{Q}_l$ is the set of queries from labeled segments of the base data set, $\mathcal{Q}_u$ is the set of queries from unlabeled segments, $\mathcal{C}_l$ is the set of prototypes based on labeled support samples, and $\mathcal{C}_u$ is the set of prototypes based on unlabeled support samples.

$$p_\phi(y = k \mid x) = \frac{\exp\left(-d\left(f_\phi(x), c_k\right)\right)}{\sum_{k'} \exp\left(-d\left(f_\phi(x), c_{k'}\right)\right)} \tag{1}$$

$$L(\phi) = -\log p_\phi(y = k \mid x) \tag{2}$$

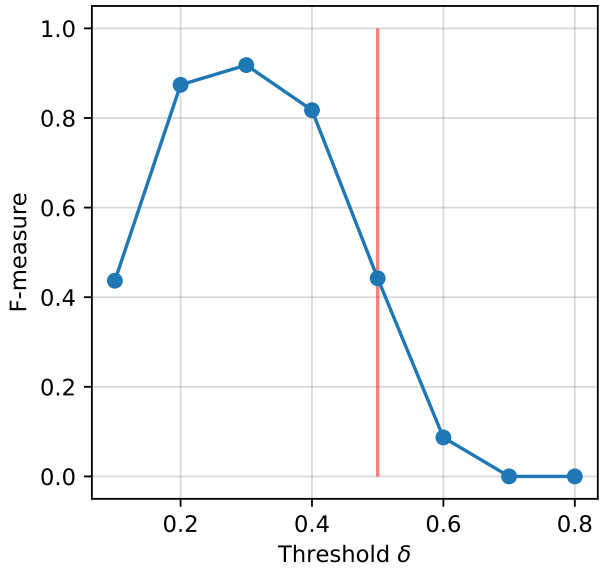

Figure 2: Impact on F-measure for different values of the threshold value used in audio segment classification at test time for one audio file in the validation set, *R4_cleaned recording_TEL_20-10-17.wav*

$$L(\phi) = -\log\left(\mathbb{1}(x \in \mathcal{Q}_l)\frac{\exp\left(-d\left(f_\phi(x), c_k\right)\right)}{\sum_{k' \in \mathcal{C}_l \cup \mathcal{C}_u} \exp\left(-d\left(f_\phi(x), c_{k'}\right)\right)} + \mathbb{1}(x \in \mathcal{Q}_u)\frac{\exp\left(-d\left(f_\phi(x), c_k\right)\right)}{\sum_{k' \in \mathcal{C}_l \cup \{k\}} \exp\left(-d\left(f_\phi(x), c_{k'}\right)\right)}\right) \tag{3}$$

The loss (3) has two inner components, one active for labeled query points from the base data set and one active for query points for which we have no label information.

6. PROBABILITY CALIBRATION

At test time an audio segment is classified as either containing the event of interest or being background. This is based on the representation of the segment in the embedding space and the distances of this point to the positive and negative prototypes. Let $d(f_\theta(x), c_p)$ and $d(f_\theta(x), c_n)$, respectively, denote these distances, where c_p and c_n represent the prototypes in the embedding space. A softmax function is then applied to the distances, and the value representing the query's probability to contain an event is used and compared to a threshold. This is the same procedure used in the deep learning baseline. See (4) for a description of the classification function, where $p(y = 1|x)$ is the softmax over the distances and δ is a threshold.

$$\mathbb{1}\left[p(y = 1|x) > \delta\right] \tag{4}$$

What the threshold δ should be for best downstream performance is not obvious. However, not using the default value of $= 0.5$ could lead do better predictions in the downstream task, as shown in Figure 2, where F-measures are calculated for a number of different threshold values for one of the few-shot tasks in the validation set.

How to find the threshold which yields the best result is not trivial given the limited amount of training data for the few-shot tasks. Given the event length for an audio file in the validation set, the pre-processing results in a set $\mathcal{X}$ of positives (spectrograms of audio segments) of varying sizes. For example, audio files with very short events typically result in the same amount of spectrograms as annotations, for this challenge five. However, for audio files with longer events we get $|\mathcal{X}| > 5$. In addition to this set we also have a set of negatives $\mathcal{N}$; these are the parts of the audio file which are in between the five first annotations and serve as hard negatives. Given these sets we fit a logistic regression model using scikit-learn [3]. This is done by first uniformly sampling half of the data points in $\mathcal{X}$ to serve as a positive prototype $\mathcal{X}_p$, while the remainder will be used as training data $\mathcal{X}_t^p$. This is followed by sampling $|\mathcal{X}_t^p|$ data points from $\mathcal{N}$ with replacement to achieve a balanced learning problem; we denote this set as $\mathcal{X}_t^n$. Note that the negative prototype based on the audio file is used here and not the one based on $\mathcal{N}$. Given the prototypes and these data points we get probabilities of a segment belonging to the positive class using a softmax, as described in Section 6. Given the fit of the logistic regression model on this one dimensional data we get a threshold which yields the highest accuracy on the training data. This procedure was repeated and an average taken.

With δ^* denoting the learned threshold, we finally set the threshold for a given audio file according to (5), where l is given in (6). This is done because the fitting resulted in very poor thresholds for audio files with extremely short event lengths, i.e., audio files which are associated with small $\mathcal{X}$ sets.

$$\delta_{\text{final}} = (1 - l) * 0.5 + l * \delta^* \tag{5}$$

$$l = \tanh\left(0.05 * (|\mathcal{X}| - 5)\right) \tag{6}$$

7. RESULTS

The results for the four different submissions on the validation set can be seen in Table 1. The submission willbo_supervised_2, which uses an adaptive threshold at test time, achieves the best total F-measure of 57.55%.

Table 1: F-measure for the subsets and for the whole validation set for the four submissions.

Method	PB	ME	HB	Total
willbo_supervised_1	39.3	75	51.02	51.39
willbo_supervised_2	**40.71**	77.31	**68.36**	**57.55**
willbo_semi_1	35.29	75.79	56.87	50.78
willbo_semi_2	29.12	**77.78**	65.98	47.94

8. REFERENCES

[1] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," 2017. [Online]. Available: https://arxiv.org/abs/1703.05175

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.