ITERATIVE KNOWLEDGE DISTILLATION IN R-CNNS FOR WEAKLY-LABELED SEMI-SUPERVISED SOUND EVENT DETECTION

Technical Report

Khaled Koutini, Hamid Eghbal-zadeh[*]*, Gerhard Widmer*

Institute of Computational Perception (CP-JKU),
Johannes Kepler University Linz, Austria
khaled.koutini@jku.at

ABSTRACT

In this technical report, we present our approach used for the CP-JKU submission in Task 4 of the DCASE-2018 Challenge. We propose a novel iterative knowledge distillation technique for weakly-labeled semi-supervised event detection using neural networks, specifically Recurrent Convolutional Neural Networks (R-CNNs). R-CNNs are used to tag the unlabeled data and predict strong labels. Further, we use the R-CNN strong pseudo-labels on the training datasets and train new models after applying label-smoothing techniques on the strong pseudo-labels. Our proposed approach significantly improved the performance of the baseline, achieving the event-based f-measure of 40.86% compared to 15.11% event-based f-measure of the baseline in the provided test set from the development dataset.

Index Terms— Weakly-labeled, Semi-supervised, Knowledge Distillation, Recurrent Neural Network, Convolutional Neural Network

1. INTRODUCTION

Motivated by the release of Audioset [1], the task of predicting strong labels using models trained on weakly-labeled audio data was introduced in the DCASE-2017 challenge (task 4) [2]. However, in DCASE-2018, the task has changed and transformed into a semi-supervised task which adds another dimension of complexity to this challenge. By leaving the majority of the training data unlabeled [3], the organizers motivated the participants to leverage the large sets of unlabeled data in a semi-supervised manner in order to improve the performance of their systems. Another important change compared to DCASE-2017 is the evaluation metric, that is changed from segment-based evaluation to event based evaluation. In DCASE-2018 task4, the submissions will be evaluated by the macro average of class-wise *event-based* F1-scores (explained in Section 3.3). The new evaluation metric introduces new challenges to the task, since the systems need to predict the onsets and offsets of the events very accurately. In other word, unlike DCASE-2017, events that are partially detected – with inaccurate onsets and offsets– do not improve the performance based on the new evaluation metric, but rather worsen it, as it will get evaluated as both a false positive and a false negative [3]. In this paper, we propose a novel approach to overcome the difficulties of this new task by leveraging the unlabeled data via an iterative knowledge distillation in neural networks. We show that using our method, the performance of a Convolutional Recurrent Neural Network (R-CNN) can be significant improved. We provide experimental results on DCASE-2018 task 4 dataset and compare it with the baselines we used.

2. THE PROPOSED APPROACH

In this section, we detail the key components of our proposed iterative knowledge distillation method.

2.1. Proposed Approach for Audio Tagging

We train an R-CNN on the weakly-labeled dataset and predicted pseudo-weak-labels for both in-domain and out-of-domain sets. Table 2 shows the configuration of the layers of the model.

2.2. The Proposed Approach for Strong Label Prediction

We follow a multi-pass strategy to get our final predictions, by iteratively predicting pseudo-strong-labels for the labeled, in-domain and out-of-domain sets, and retraining new models on those new predictions.

2.2.1. The First Pass

We trained a recurrent convolutional neural network with the same architecture that was used for tagging (Table 2). However, the network is not only trained on the provided labels of the labeled set, but also on the predicted pseudo labels for both the in-domain and out-of-domain sets. The result of the first pass are strong labels for the labeled, in-domain and out-of-domain sets. These labels are presented in the form of frame-level probabilities for every audio clip.

2.2.2. The Second Pass

In the second pass, we smooth the current predicted pseudo-strong labels using median/Gaussian filters and we train new models on them. We observed that the performance of the models varies among different classes. We achieved better performances in some classes using a deep model (Table 2), while for other classes shallow models (Table 3) performed better. In addition, using median smoothing with or without Gaussian smoothing resulted in varying performances for different classes.

[*] The second author which was part of the organization team, confirms that all the system development for this submission was done only on the development data, without using any knowledge from the evaluation data.

Table 1: F-score results per class for each submission. The average is calculated class-wise (macro-average) [4].

Submission.	Average	Alarm	Blender	Cat	Dishes	Dog	Electric..	Frying	Runnin..	Speech	Vacuum..
Baseline	15.11	3.8	12.2	1.6	0.0	3.8	36.7	34.4	10.2	0.0	48.4
Koutini_JKU_task4_1	40.86	49.3	40.0	50.0	18.1	25.7	44.1	43.5	31.0	49.9	57.1
Koutini_JKU_task4_2	40.23	49.3	39.6	50.0	18.1	25.7	44.1	41.7	30.4	49.0	54.5
Koutini_JKU_task4_3	39.26	49.3	39.6	49.4	17.8	25.5	41.9	38.5	29.8	48.4	52.3
Koutini_JKU_task4_4	35.63	47.9	37.3	47.9	14.5	23.9	35.8	36.1	26.7	46.0	40.3

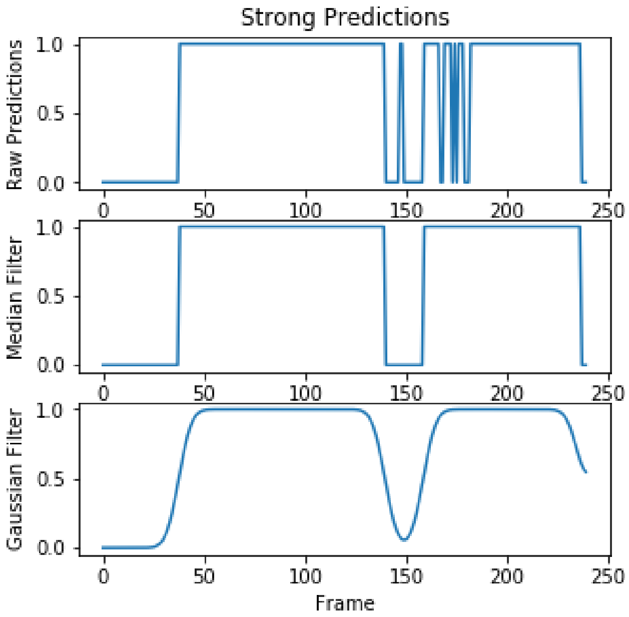

Figure 1: Example of strong predictions before/after smoothing.

2.2.3. *Model selection*

We train multiple models with/without smoothing. Then, we select the best trained model for each class to predict new pseudo-strong-labels for the respected class for the labeled, in-domain and out-of-domain sets. Using these new prediction, we iteratively repeated the second pass (Figure 2).

2.2.4. *Smoothing for Strong Prediction*

The strong predictions of our models trained only on weakly-labeled data tend to be noisy. Therefore, we smooth those predictions using median and Gaussian filters (Figure 1). We then use these smoothed probabilities for retraining the network in the next pass as explained in Section 2.2.

3. EXPERIMENTS AND RESULTS

3.1. Dataset

The dataset is split into a training set, a test set and an evaluation set [3]. The training set contains three subsets, a labeled set, an unlabeled-in-domain set and an unlabeled-out-of-domain set. In this paper, they are referred to as labeled, in-domain, out-of-domain respectively. The test set contains 288 strongly labeled audio clips.

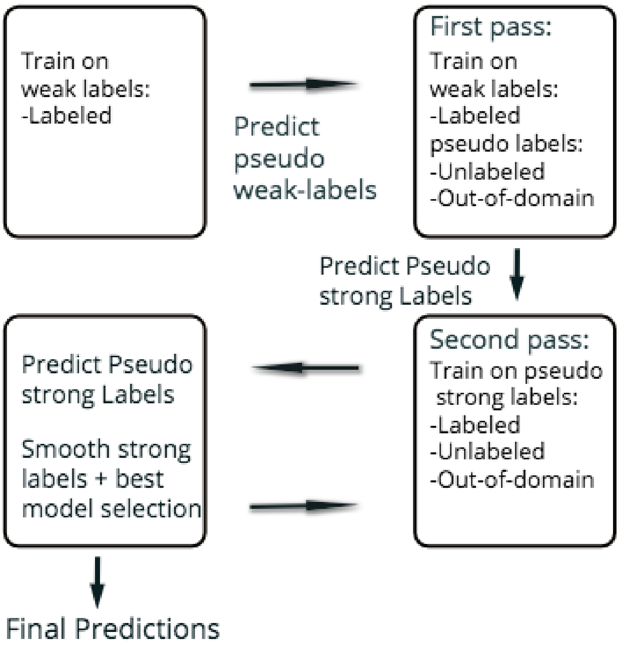

Figure 2: The proposed knowledge distillation framework for RC-NNs.

The evaluation set consist of 880 audio clips, for which our system predicted strong labels for the challenge submission.

3.2. Features Extraction

We use log-scaled Mel-bands spectrograms as an input for all our models. We extracted 64 Mel bands from 64 ms frames with 22.5 ms overlap using Librosa [5]. That resulted in an input size of 240 × 64 for our models.

3.3. Evaluation Metric

The evaluation metric for the task is the event-based F-score [4]. The predicted events are compared with a reference event list, by comparing the onset and the offset of the predicted event with the overlapping reference event. The predicted event is considered correctly detected (true positive), if it's onset is within 200 ms collar of the reference event onset and its offset is within 200 ms or 20% of the event length collar around the reference offset. If a reference event has no matching predicted event, it is considered a false negative. If the predicted event doesn't match any reference event, it is considered a false positive. Furthermore, if the system partially predicted an event without accurately detecting its onset and offset,

Table 2: Proposed deep architecture for predicting strong labels and audio tagging. BN: Batch normalization, BIAS: Model uses bias with no batch normalization, ReLu: Rectified Linear activation function

<table>
<tr><th colspan="2">Input 240 × 64</th></tr>
<tr><td colspan="2">2 × 2 Conv(pad-1, stride-1)-64-BN-ReLu
2 × 2 Conv(pad-1, stride-1)-64-BN-ReLu
1 × 2 Max-Pooling</td></tr>
<tr><td colspan="2">2 × 2 Conv(pad-1, stride-1)-64-BN-ReLu
2 × 2 Conv(pad-1, stride-1)-64-BN-ReLu
1 × 2 Max-Pooling</td></tr>
<tr><td colspan="2">2 × 2 Conv(pad-1, stride-1)-64-BN-ReLu
2 × 2 Conv(pad-1, stride-1)-64-BN-ReLu
1 × 2 Max-Pooling</td></tr>
<tr><td colspan="2">2 × 2 Conv(pad-1, stride-1)-64-BN-ReLu
2 × 2 Conv(pad-1, stride-1)-64-BN-ReLu
1 × 2 Max-Pooling</td></tr>
<tr><td colspan="2">1 × 1 Conv(pad-1, stride-1)-256-BIAS-ReLu
1 × 4 Max-Pooling</td></tr>
<tr><td colspan="2">Bi-directional SRU 128 hidden units</td></tr>
<tr><td colspan="2">1 × 1 Conv(pad-1, stride-1)-10-BIAS-Sigmoid
Output 240 × 10</td></tr>
<tr><td>(Strong predictions)
Output 240 × 10</td><td>(Weak-label training and tagging)
Global-Average-Pooling
Output 10</td></tr>
</table>

Table 3: Proposed shallow architecture for predicting strong labels. Similar to the baseline [3]. BN: Batch normalization, BIAS: Model uses bias with no batch normalization, ReLu: Rectified Linear activation function

<table>
<tr><th colspan="2">Input 240 × 64</th></tr>
<tr><td colspan="2">3 × 3 Conv(pad-1, stride-1)-64-BN-ReLu
1 × 4 Max-Pooling</td></tr>
<tr><td colspan="2">3 × 3 Conv(pad-1, stride-1)-64-BN-ReLu
1 × 4 Max-Pooling</td></tr>
<tr><td colspan="2">3 × 3 Conv(pad-1, stride-1)-64-BN-ReLu
1 × 4 Max-Pooling</td></tr>
<tr><td colspan="2">Bi-directional SRU 128 hidden units</td></tr>
<tr><td colspan="2">1 × 1 Conv(pad-1, stride-1)-10-BIAS-Sigmoid
Output 240 × 10</td></tr>
<tr><td>(Strong predictions)
Output 240 × 10</td><td>(Weak-label training and tagging)
Global-Average-Pooling
Output 10</td></tr>
</table>

Table 4: The performance of our approach compared to the baseline system [3]. Note that we re-ran the baseline on our machines, hence the slight difference from the reported values in [3].

	F1	Precision	Recall
Baseline	15.11	14.20	17.80
Koutini_JKU_task4_1	40.86	40.21	44.42
Koutini_JKU_task4_2	40.23	39.54	44.02
Koutini_JKU_task4_3	39.26	38.24	44.12
Koutini_JKU_task4_4	35.63	32.95	43.42

it will be penalized twice, as a false positive and a false negative. Equation (1) shows the calculation of the F-score for each class [3].

$$F_c = \frac{2.TP_c}{2.TP_c + FP_c + FN_c}, \qquad (1)$$

Where F_c, TP_c, FP_c, FN_c are the F-score, true positives, false positives, false negatives of the class c respectively. The final evaluation metric the average of the F-score for all the classes.

3.4. Results

We did 4 final submissions for the challenge using the probabilities of the same system. However, the submission are different in the final smoothing window length and the thresholds which were selected based on the results on the validation set.

Table 1 shows the class-wise F-score results for all the 4 submitted systems compared to those of the baseline system.

Table 4 shows the final macro-averaged event-based evaluation results on the test set compared to the baseline system.

4. ACKNOWLEDGMENT

This work has been supported by the COMET-K2 Center for Symbiotic Mechatronics of the Linz Center of Mechatronics (LCM) funded by the Austrian federal government and the federal state of Upper Austria, and additionally funded by the Austrian Ministries BMVIT and BMWFW, and the Province of Upper Austria, via the COMET Center SCCH.

5. REFERENCES

[1] J. F. Gemmeke, D. P. 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 *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 776–780.

[2] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, "Dcase 2017 challenge setup: Tasks, datasets and baseline system," in *DCASE 2017-Workshop on Detection and Classification of Acoustic Scenes and Events*, 2017.

[3] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. Parag Shah, "Large-Scale Weakly Labeled Semi-Supervised Sound Event Detection in Domestic Environments," July 2018, submitted to DCASE2018 Workshop. [Online]. Available: https://hal.inria.fr/hal-01850270

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

[5] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, 2015, pp. 18–25.