DCASE 2022 TASK 4 TECHNICAL REPORT

Technical Report

Kornel Błąkała, Olaf Sikorski

Samsung R&D Intsitute Poland
{k.blakala, o.sikorski}@samsung.com
Warsaw, Poland

ABSTRACT

This paper describes our solution for Task 4 of the 2022 edition of the Detection and Classification of Acoustic Scenes and Events competition. Our solution practically consists of two specialised systems that excel in either of the two scenarios in the challenge. Both utilise the CRNN model architecture and mean-teacher training setup proposed in the baseline solution. The modifications that they share are the replacement of the CNN extractor with a ResNet-18 architecture and the reduction of the FFT window from 2048 to 1024 samples. The systems diverge in four aspects: the set of augmentations selected and whether they use any additional techniques during training. For Scenario 1 we observed improvement when using pitch shift, while all other data augmentation methods resulted in lower PSDS. On the other hand, Scenario 2 benefited greatly from spectrogram time warping and adding brown noise. Further improvement on Scenario 2 was achieved by replacing attention with mean aggregation for weak predictions, incorporating per-frame embeddings from Audio Spectrogram Transformer (AST) and injecting Gaussian noise between teacher and student during consistency loss calculation. Curiously, these modifications diminished performance on Scenario 1. The system specialising in Scenario 1 scored $[0.3743,\ 0.5826]$ and the system specialising in Scenario 2 scored $[0.0701,\ 0.7938]$ in $[PSDS1,\ PSDS2]$ respectively.

***Index Terms*—** Sound event detection, Self-supervised, Student-teacher

1. PROBLEM OUTLINE

Task 4 of the Detection and Classification of Acoustic Scenes and Events (DCASE) competition challenges the participants with a Sound Event Detection (SED) problem, where the objective is to detect one of 10 possible sounds in 10-second-long audio recordings and provide their temporal location within the sample. While SED is not a new task in the field of audio research, it is notoriously hard due to the lack of large quantities of strongly-labelled data, that is audio recordings accompanied by onset-offset timestamps indicating the presence of a given sound at that time. The setup of this competition naturally reflects that fact and organisers provide participants with three distinct types of datasets: strongly-labelled, weakly-labelled (indicating the presence, but not the temporal location) and unlabelled. Additionally, the entire strongly-labelled dataset was generated synthetically using Scaper [1], which is a tool that overlays labelled sound events over each other and a background to produce strongly-labelled samples.

2. METHODS

2.1. Network architecture

SED problems are commonly modelled using a Convolutional Recurrent Neural Network[2] model. Such architectures combine the feature extraction ability of convolutional networks and the bias towards learning sequential patterns of recurrent networks. These aspects make them well-suited for tackling audio segmentation tasks where data is inherently a temporal sequence.

The architectures we explored during this competition were all CRNN's. We tried several convolutional feature extractors in place of the default CNN suggested in the baseline and found the 18-layer ResNet (ResNet-18)[3] to yield the best results, albeit with modest improvements. Meanwhile, the recurrent section remained a two-layer bidirectional GRU.

2.2. Training framework

Given the variety of label types (or lack of thereof) the problem cannot be effectively solved using pure supervised learning, but invites for the application of semi-supervised learning techniques that take full-advantage of the unlabelled and weakly labelled data. The baseline system implements a mean-teacher framework[4] which is one of many possible approaches in semi-supervised learning[5]. The core idea behind it is to teach two models simultaneously: a student model that is updated from loss via backpropagation and a teacher model that has its weights calculated as an Exponential Moving Average (EMA) of the weights of successive generations of students (hence "mean" in the name).

In mean-teacher the loss consists of two components: consistency loss and cross-entropy loss. Cross-entropy loss requires that the samples come with strong or weak labels, while consistency loss is calculated for all samples as a Mean Squared Error (MSE) between the teacher's and the student's prediction on a given sample. Consistency loss takes advantage of the assumption that real-world data-to-label mappings are smooth and clustered. Smoothness assumes that if two data points are close together in the sample space then so should their corresponding labels. The cluster assumption postulates that data points of the same class form regions of high density in the sample space that are separated by regions of low density. This allows the algorithm to use consistency to meaningfully extend beyond the region covered by labelled data and learn from unlabelled samples to improve generalisation.

We found mean-teacher satisfactory for the reasons mentioned above as well as due to the fact that averaging students produced an extremely robust teacher model that exhibited very little vari-

ance between epochs. The teacher model consistently outperformed the student from the early stages of training, a phenomenon documented in the literature[4], while improving nearly perfectly monotonically over training steps. Those facts greatly simplified the selection of models from training and reduced variance between runs.

2.3. Gaussian noise injection

Improving model robustness against against random perturbation is intuitively a way of improving performance of any classifier and has been shown to improve the generalisation of neural networks in particular[6], including in the context of semi-supervised learning[7]. This is extremely relevant for mean-teacher, which relies on consistency in incorporating unlabelled data into training. In fact, previous researchers have investigated ways of injecting designed perturbation into the system in order to smoothen the output[8].

In mean-teacher noise injection can be achieved by overlaying a selected perturbation over a sample right before inference through teacher or student. It might make more intuitive sense to perturb the student, since the teacher is responsible for generating a ground truth that we intend the student to match. However, our consistency loss is symmetric, since we use MSE, so it makes little difference in practice. Indeed, through all our experiments we found that the results were very similar irrespective of the injection point. The best-performing model on scenario 2 (System ID: 2) used the teacher as the injection point. It is theoretically possible to use both points simultaneously, yet we did not experiment with it.

The perturbation we selected was simple Gaussian noise applied to the log-mel spectrogram. We scaled it per sample so that the perturbation of a pixel in sample S was at most:

$$pert_{max} = \epsilon * (max(S) - min(S))$$

where $\epsilon = 0.1$ was empirically found optimal.

We have also experimented with more sophisticated perturbations inspired by Virtual Adversarial Training (VAT)[8]. In VAT, Miyato postulates that changing the sample in a direction in the input space that leads to greatest variation in the output allows more efficient output smoothing, as it addresses the anisotropy of the model. In other words, it assures the model is equally sensitive to perturbation in all directions by prioritising the most vulnerable ones first. We attempted to calculate the virtual adversarial perturbation using a Fast Gradient Sign Method (FGSM)[9] and then inject it during consistency loss calculation. Unfortunately, the results were not as good as for simple Gaussian noise, so this technique was not used in the submitted system.

2.4. Augmentation methods

We also used 3 simple augmentation methods: pitch shifting, time warping and adding a brown noise. For pitch shifting, the audio samples were translated by a random number of semitones from -4 to 4 using the torch-audiomentations implementation[1]. Time warping is done by choosing a point in time on spectrogram, and squeezing spectrogram on one side, and stretching on the other side by a random factor [10]. Adding brown noise is done with function *AddColoredNoise* from torch-audiomentations library.

[1]https://github.com/asteroid-team/torch-audiomentations

3. EXPERIMENTS

3.1. Feature extraction

Before feeding the data to the neural net we transform it into log-mel spectrogram. Parameters of log-mel spectrogram transformation are as follows: windows size of 1024, hop size of 256, 128 mel bins and Hamming windows. The spectogram shape is 626×128 for a 10 seconds recording.

3.2. Pipeline and training settings

All experiments are conducted using DCASE 2022 baseline environment[2]. We implemented proposed methods to be compatible with this repository.

4. RESULTS AND ANALYSIS

We performed multiple experiments to find the best models for Scenario 1 and Scenario 2. Experiments that lead us to the submitted systems 1 and 2 are shown in Table 1 and Table 2, sequentially adding new modifications to the training.

For Scenario 1 we only modified baseline with smaller window size, resnet backbone and pitch shift augmentation. The window size modification increased PSDS1 by almost 3 percentage points, pitch shift increased the PSDS1 by 1 pp. The usage of resnet backbone did not change PSDS1 that much, but we decided to continue with it. All modifications also improved PSDS2.

As for Scenario 2, we used more modifications that greatly boosted PSDS2, but some of them decreased PSDS1 to the point that this system would not be anywhere near baseline performance. All modifications and the resulting PSDS1 and PSDS2 are shown in the Table 2. Weak predictions as a mean over time axis of strong predictions increased the PSDS2 by 6.5 pp, but at the same time decreased PSDS1 by over 20 pp. The reasoning behind simple mean instead of the attention in baseline architecture was to transfer more of the class information contained in weak labeled real data. We also tried using max function instead of mean function, to focus on the most visible event in the recording, but we got much worse results for PSDS2. Similarly, time warping resulted in higher PSDS2, but lowered PSDS1. That is the effect of time warping only audio, not the strong label matrix. While it resulted in a mismatch in localisation, it did not diminish PSDS2 since it required a mere 0.1 intersection. We tried time warping both the label matrix and audio at the same time, but it resulted in a far smaller boost in PSDS2. Another modification that decreased first scenario metric was usage of bigger ResNet, that halved time resolution likely further lowered the resulting PSDS1 score.

Table 1: Path to best Scenario 1 architecture

System ID	Performance Metrics	
	PSDS1	PSDS2
baseline	0.336	0.536
+ 1024fft	0.364	0.554
+ resnet	0.364	0.554
+ pitch shift	0.374	0.583

[2]https://github.com/DCASE-REPO/DESED_task

Table 2: Path to best Scenario 2 architecture

System ID	Performance Metrics	
	PSDS1	PSDS2
baseline	0.336	0.536
+ 1024fft	0.364	0.554
+ resnet bigger	0.306	0.600
+ brown noise	0.313	0.606
+ time warp	0.289	0.675
+ mean weak predictions	0.052	0.740
+ noise injection	0.056	0.751
+ AST embeddings	0.064	0.794

Table 3: Submitted systems and PSDS scores for both scenarios

System ID	Performance Metrics	
	PSDS1	PSDS2
1.	0.37433	0.58255
2.	0.06401	0.79380
3.	0.34105	0.59564

5. SUBMISSIONS

Submitted systems are shown in Table 3, we submitted 3 systems. First is our best in PSDS1, second is our best in PSDS2, and the last one is the system trained without any external data. Our team-wise scores on validation dataset are PSDS1: 0.37433 and PSDS2: 0.79380.

6. REFERENCES

[1] J. Salamon, D. MacConnell, M. Cartwright, P. Q. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 344–348, 2017.

[2] E. Çakir, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, pp. 1291–1303, 2017.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[4] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in neural information processing systems*, vol. 30, 2017.

[5] Y. Ouali, C. Hudelot, and M. Tami, "An overview of deep semi-supervised learning," *arXiv preprint arXiv:2006.05278*, 2020.

[6] C. M. Bishop, "Training with noise is equivalent to tikhonov regularization," *Neural computation*, vol. 7, no. 1, pp. 108–116, 1995.

[7] M. Sajjadi, M. Javanmardi, and T. Tasdizen, "Regularization with stochastic transformations and perturbations for deep semi-supervised learning," *Advances in neural information processing systems*, vol. 29, 2016.

[8] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[10] Y. Hwang, H. Cho, H. Yang, D.-O. Won, I. Oh, and S.-W. Lee, "Mel-spectrogram augmentation for sequence to sequence voice conversion," 2020. [Online]. Available: https://arxiv.org/abs/2001.01401