ACOUSTIC SCENE CLASSIFICATION WITH REJECT OPTION BASED ON RESNETS

Bernhard Lehner[1,2] *Khaled Koutini*[2] *Christopher Schwarzlmüller*[1]
Thomas Gallien[1] *Gerhard Widmer*[2]

[1]Silicon Austria Labs (SAL)
[2]Institute of Computational Perception,
Johannes Kepler University Linz, Austria
bernhard.lehner@silicon-austria.com

ABSTRACT

This technical report describes the submissions from the SAL/CP JKU team for Task 1 - Subtask C (classification on data that includes classes not encountered in the training data) of the DCASE-2019 challenge. Our method uses a ResNet variant specifically adapted to be used along with spectrograms in the context of Acoustic Scene Classification (ASC). The reject option is based on the logit values of the same networks. We do not use any of the provided external data sets, and perform data augmentation only with the *mix-up* technique [1]. The result of our experiments is a system that achieves classification accuracies of up to around 60% on the public Kaggle-Leaderboard. This is an improvement of around 14 percentage points compared to the official DCASE 2019 baseline.

Index Terms— audio scene classification, convolutional neural networks, deep learning, reject option

1. INTRODUCTION

In this report, we describe our approach to deal with a problem that is present in many real-world machine learning tasks: the *reject option*, also known as *selective prediction* or *open set classification*. It is the problem of developing models that "know what they don't know", and extremely challenging, since the amount of examples of unknown (i.e., not used in training) classes is infinite, and yet such examples need to be identified as such.

In this specific case, we want a model capable to distinguish 10 different audio scenes (e.g., *bus*, *airport*, or *park*). At test-time – in addition to these 10 classes – several audio scenes that are not part of the training data are presented to the system, and it is supposed to classify them as *unknown*. In the development dataset provided by the DCASE challenge organisers, there are examples of four different audio scenes that are not part of the training data, hence *unknown*.

However, there is no guarantee that in the official evaluation data the *unknown* examples will be recordings from the same four audio scenes. Therefore, one has to be very careful not to overestimate the reliability of the validation results. Furthermore, simply including the examples of the *unknown* classes in the training to be recognised as an 11^{th} class is not very promising, and most likely will lead to overfitting.

2. CONVOLUTIONAL NEURAL NETWORKS FOR ASC

In this section we describe the neural network architectures as well as the optimisation strategies used for training our ASC networks.

2.1. Network Architecture

ResNet exhibits residual connections between its convolutional layers in order to alleviate the vanishing gradient problem of very deep architectures [2]. For image processing tasks, ResNet is one of the top-performing architectures, especially compared to less recent architectures like AlexNet [3] or VGG [4].

However, this improvement does not necessarily translate to the audio processing domain, where VGG-ish architectures dominated the submissions of the previous DCASE 2018 [5], [6]. Usually, ResNet variants tend to overfit, unless very large amounts of training data are available – which is often not the case regarding audio processing tasks. Recently, it was discovered that the receptive field (RF) – essentially, what the network effectively sees from the input – is partly responsible for this.

In [7], it was shown that the RF acts as a regulariser, and can be adapted in order to yield a variant of the ResNet architecture that surpasses the performance of a VGG given the same input in the form of spectrograms. Such a specifically adapted (i.e., RF regularised) version of a ResNet is our starting point, and we train it to classify the given 10 classes. In Table 1, the architecture is described in more detail. More detailed information about the motivation behind our ResNet variant can be found in [7] and [8].

Table 1: Modified ResNet architecture

RB Number	RB Config
	Input 5×5 stride=2
1	$3 \times 3, 1 \times 1$, P
2	$3 \times 3, 3 \times 3$, P
3	$3 \times 3, 3 \times 3$
4	$3 \times 3, 1 \times 1$, P
5	$1 \times 1, 1 \times 1$
6	$1 \times 1, 1 \times 1$
7	$1 \times 1, 1 \times 1$
8	$1 \times 1, 1 \times 1$
9	$1 \times 1, 1 \times 1$
10	$1 \times 1, 1 \times 1$
11	$1 \times 1, 1 \times 1$
12	$1 \times 1, 1 \times 1$

RB: Residual Block, P: 2×2 max pooling after the block.
RB number 1-4 have 128 channels,
RB number 5-8 have 256 channels,
RB number 9-12 have 512 channels.

2.2. Frequency-Aware CNNs

Since we are using fully convolutional networks, learned filters are agnostic to frequency information of the feature maps. In other words, the spectrograms and feature maps can be rolled over in both the time and frequency dimension with a minor impact on the network predictions. One way to make filters more specialised in certain frequencies is to concatenate a new channel containing the frequency information of each spatial pixel to each feature map. The value of the pixel $V(f,t)$ with the spatial index (f,t) in the new channel is calculated as follows.

$$V(f,t) = f/F \tag{1}$$

where F is the frequency dimension size of the feature map. This additional channel gives convolutional filters a frequency context.

2.3. Network Optimisation

We use Adam [9] with a specific scheduling as follows. First, we start training with a learning rate of 1×10^{-4}. From epoch 50 until 250, the learning rate decays from 1×10^{-4} to the minimum learning rate 5×10^{-6} in a linear fashion. Afterwards, we keep training for an additional 100 epochs with the minimum learning rate in a similar setup as in [7].

2.4. Data Preparation

For data preprocessing we resample the audio signals to 22050 Hz, and compute a single-channel (mono) Short Time Fourier Transform (STFT) using 2048-sample windows and a hop-size of 512 samples. Next, we apply a perceptual weighting to the individual frequency bands of the power spectrogram [10][1]. As a last step, we apply a mel-scaled filterbank yielding 431-frame spectrograms with 256 frequency bins per data point.

We normalise the examples to have zero-mean and unit standard deviation along the individual frequency bins according to the training data. We do not use the examples representing *unknown* classes for training, hence reducing the number of training examples used for training the network from 10290 down to 9185.

2.5. Data Augmentation

We use the *mix-up* technique [11] on the training data as the only means of data augmentation. From our experience, this augmentation method has the biggest impact on performance and generalisation capability, compared to other methods like e.g., pitch shifting or adding noise.

2.6. Reject Option

For the identification of *unknown* audio scenes we use the single maximum logit value (i.e., the input to the softmax layer) corresponding to the most likely of the 10 known classes, and apply a threshold. An example is then considered *unknown* as soon as the value of this logit is below the threshold. The threshold could in principle be determined according to the best results of the validation data. However, these results might not be reliable enough due to a lack of proper variety. Therefore, we suggest to apply a different – higher – threshold, even though this means that the validation results will decrease.

[1] `librosa.core.perceptual_weighting`

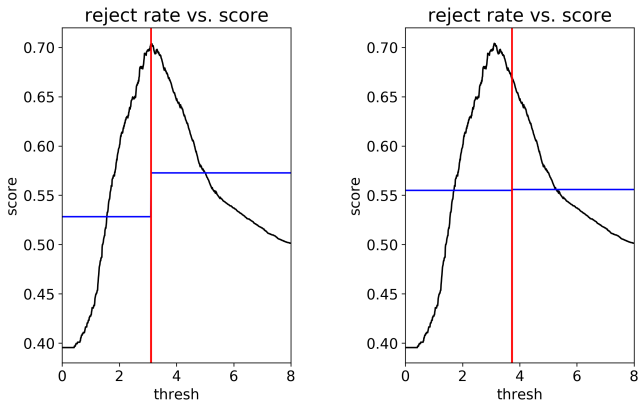

Figure 1: The total score as a function of the threshold for rejects. We consider relying too much on the validation results dangerous. Left plot: the average scores (blue) from below and above optimum threshold (red) according to the validation results; right: increased threshold leads to equal average scores on both sides of the threshold and we consider this better suited for unseen data.

The reason for that is the way the total score for this task is computed: it is the weighted average of the known classes and unknown class accuracies. This specific metric favors aggressive rejection, which might not be clear at first. In Figure 1, we can see the total score as a function of the threshold used for the reject option. As the threshold increases, the rejection rate increases as well. As can be seen, the total score when nothing gets rejected (i.e., $threshold = 0$) is around 0.4. On the other hand, when *every* example gets rejected (i.e., $threshold > 8$), the score is exactly 0.5. The blue lines correspond to the average score below and above the chosen threshold (red line), respectively. In the left plot, we have chosen the threshold according to the maximum score of the validation data. As can be seen, the average scores from below and above threshold are different, something that we consider an indicator of biased thresholding. In the right plot, we increased the threshold, which leads to equal average scores on both sides of the threshold, something that we think is better suited to deal with unseen data.

2.7. Ensemble

In order to improve robustness and generalisation of our system, we create an ensemble of four models, carefully selected according to the 10-class accuracy of the validation data. First, we average the logits of the four models, and select the label that corresponds to the maximum logit value.

These prediction results are then shared across all the systems for further processing regarding the reject option. For the reject option, we implemented four different strategies as follows.

- SAL_CNN_1: all four models have their own thresholds according to the best validation result. Reject if three or more models vote for reject. Of all methods, this yields the maximum score on the validation data, but for reasons discussed in Section 2.6 we don't consider this result reliable and expect the lowest generalisation capability of all methods.
- SAL_CNN_2: all four models have their own thresholds, but this time increased by 10% compared to the previously used thresholds. Reject if one or more models vote for reject.

	Validation			Leaderboard		Challenge		
	known	unknown	total	public	private	known	unknown	total
SAL_CNN_1	.608	.846	.727	.553	.558	n/a	n/a	n/a
SAL_CNN_2	.371	.956	.664	.600	.608	n/a	n/a	n/a
SAL_CNN_3	.294	.977	.635	.605	.608	n/a	n/a	n/a
SAL_CNN_4	.313	.968	.641	.597	.612	n/a	n/a	n/a
DCASE_BL	.542	.431	.487	.467	.442	n/a	n/a	n/a

Table 2: Results of the four submitted methods. All methods share the same 10-class results, and only differ in terms of their rejection strategy. SAL_CNN_1: reject if the majority votes for rejection; SAL_CNN_2: 10% increased threshold for single model rejection, reject if one or more vote for rejection; SAL_CNN_3: 20% increased threshold for single model rejection, reject if one or more vote for rejection; SAL_CNN_4: 30% increased threshold for single model rejection, reject if two or more vote for rejection; DCASE_BL: official baseline provided by DCASE organisers [12].

- SAL_CNN_3: all four models have their own thresholds, but this time increased by 20%. Reject if one or more models vote for reject.
- SAL_CNN_4: all four models have their own thresholds, but this time increased by 30%. Reject if two or more models vote for reject.

3. RESULTS

In this section, we list our submitted systems and their respective performances on the public and private Kaggle leaderboard and the final evaluation set. Just using the vanilla ResNet without any reject option would result in a total score of 0.407 (known: 0.814, unknown: 0.0).

As can be seen in the validation data results, our four methods are quite different regarding the aggressiveness of their rejection (the score for *unknown* is equivalent to its recall). The least aggressive rejection strategy (SAL_CNN_1) yields the best results on the validation data, yet the lowest score on the leaderboard. We intentionally submitted this method in order to demonstrate the risk of overfitting when relying too much on the validation results.

The remaining three methods yield similar total scores, but the accuracies of the *known* and *unknown* are quite different. Interestingly, our most aggressive rejection strategy (SAL_CNN_3) yields the lowest score on the validation data, yet the generalisation gap compared to the leaderboard results is the smallest.

Regarding the Kaggle leaderboard, both public and private leaderboard results are quite consistent for all four methods.

4. CONCLUSION

This technical report describes our submissions for Task 1 - Subtask C of the DCASE-2019 challenge. We use a ResNet variant specifically adapted in terms of its receptive field to be used along with spectrograms. The reject option is based on the logit values of the same networks. We show the risk of tuning the reject option according to the validation data scores, and propose a more aggressive rejection, which results in lower validation scores, but seems to generalise better. We do not use any of the provided external data sets, and perform data augmentation only with the *mix-up* technique [1]. We improve around 14 percentage points upon to the official DCASE 2019 baseline [12] on the public Kaggle leaderboard.

5. ACKNOWLEDGMENT

This work has been supported by the COMET-K2 Center of the Linz Center of Mechatronics (LCM) funded by the Austrian federal government and the federal state of Upper Austria. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of a Titan Xp GPU used for this research.

6. REFERENCES

[1] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[2] 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*, pp. 770–778.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[4] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition." [Online]. Available: http://arxiv.org/abs/1409.1556

[5] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Iterative knowledge distillation in R-CNNs for weakly-labeled semi-supervised sound event detection," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pp. 173–177.

[6] M. Dorfer, B. Lehner, H. Eghbal-zadeh, C. Heindl, F. Paischer, and G. Widmer, "Acoustic Scene Classification with Fully Convolutional Neural Networks and I-Vectors." DCASE2018 Challenge.

[7] K. Koutini, H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "The Receptive Field as a Regularizer in Deep Convolutional Neural Networks for Acoustic Scene Classification," in *Proceedings of the European Signal Processing Conference (EUSIPCO)*, A Coruña, Spain, 2019.

[8] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "CP-JKU submissions to DCASE'19: Acoustic Scene Classification and Audio Tagging with Receptive-Field-Regularized CNNs," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019.

[9] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization.” [Online]. Available: http://arxiv.org/abs/1412.6980

[10] 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.

[11] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “Mixup: Beyond Empirical Risk Minimization.” [Online]. Available: http://arxiv.org/abs/1710.09412

[12] A. Mesaros, T. Heittola, and T. Virtanen, “A multi-device dataset for urban acoustic scene classification,” in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 9–13. [Online]. Available: https://arxiv.org/abs/1807.09840