MALACH23 SUBMISSION TO DCASE 2023: ACOUSTIC SCENE CLASSIFICATION WITH RECEPTIVE-FIELD REGULARIZED CONVOLUTION NEURAL NETWORKS AND STATE SPACE MODELS

Technical Report

Jonathan Greif, Noah Pichler, Christian Willdoner, David Fleischanderl

Students at Johannes Kepler University, Linz, Austria
{k12124999, k12011672, k00956037, k12007779}@students.jku.at

ABSTRACT

This report describes our approach to Task 1 of the DCASE (Detection and Classification of Acoustic Scenes and Events) Challenge [1]. To classify urban acoustic scenes through short audio samples, we experiment with Receptive Field Regularized [2] Convolutional Neural Networks - and S4 [3] Models as classifiers. To stay within the allowed model-complexity limits of the challenge, we use a Convolution Neural Network (CNN) with 13 layers plus one classification layer, and one CNN layer followed by 3 S4 Blocks, respectively. Additionally, we augment the Mel-Spectrograms, through the MixStyle [4] or Mixup [5] method. We surpass the baseline with our experiments significantly, and, in particular, the S4 model stands out due to its low number of multiply-accumulate operations.

Index Terms— Acoustic Scene Classification, Convolutional Neural Networks, S4, Receptive Field Regularization, Low-Complexity

1. INTRODUCTION

In the context of the course ”Machine Learning and Audio: a challenge” by the Institute of Computational Perception at the Johannes Kepler University Linz the authors of this report teamed up to participate in Task 1 of the DCASE Challenge 2023 [1] named ”Low-Complexity Acoustic Scene classification”. As the name suggests, its aim is the classification of short audio samples (1s, 44.1 kHz, 24-bit) into categories of typical urban locations, such as airports or shopping malls. In addition, the data can be separated by the 12 European cities in which it was recorded and the specific device that was used for recording. These recordings originated from 4 real devices (A, B, C, D) and additional 11 simulated devices (S1-11), which were generated using the data of Device A. To set a focus on the generalization capabilities of models the data from 2 of the cities and 6 of the devices (D, S7-11) is only present in the evaluation set [6]. On top of the focus on generalization capabilities, the submitted models are constrained concerning memory for the parameters (<128 KB) and MAC operations (<30 MMACs). These two constraints are also part of the evaluation metric, which can be viewed as a weighted sum of Macro Accuracy (50%), needed Memory (25%), and MAC operations (25%). In previous editions the many successful approaches built upon the features in the Mel-Spectrograms of the audio samples, which were processed through CNNs. Our approach to this task was twofold: for one we attempted to reach a good performance through experimenting with RFR-CNNs [2], while on the other hand, we were also experimenting with using an efficient S4 model [3] architecture with a low MAC and parameter count while still beating the baseline. Additionally, we used the regularization method Freq-MixStyle [4, 7].

`https://github.com/viddavue/MALACH_23_T1/`

2. DATA PREPROCESSING AND AUGMENTATION

2.1. Audio processing

The raw audio signal is down-sampled using a sampling rate of 32kHz, and the input features are extracted from the raw audio signals using a Short Time Fourier Transformation (STFT) with a hamming window of size of 2048 and an overlap of 36% for the CP-ResNet [8, 2] and the same window size and an overlap of 25% for the S4 model. We apply a Mel-scaled filter bank to end up with 256 frequency bins for both models. We also experimented with random time and frequency masking of the Mel-Spectrogram similar to SpecAugment [9], but this did not improve generalization in combination with the following two methods.

2.2. Freq-MixStyle

As in Schmid et al. [2], we experimented with Freq-MixStyle to better generalize to unseen devices. Freq-MixStyle uses an augmentation technique described in Zhou et al. [4], but instead of normalizing over the channels, we normalized over the frequencies as described in Kim et al. [7]. They show that the device style is mainly contained in the frequency statistics and by mixing those, the model learns to handle different device styles. Freq-MixStyle can be controlled by the parameter α, which specifies the shape of the Beta Distribution of drawing mixing coefficients, and p gives the probability to apply Freq-MixStyle to the batch.

2.3. Mixup

As proposed by Zhang et al. [5] we applied Mixup on the input of the S4 models as it was shown that Mixup improves generalization in general [10]. Although Freq-MixStyle should outperform Mixup [2], it seems that Freq-Mixstyle and S4 are a bad fit. In our

3. ARCHITECTURE

For our submissions, we used models from two different model classes namely a Receptive Field Regularized [11] Convolutional

Neural Network (RFR-CNN) inspired by Schmid et al. [2] and the S4 model by Gu et al. [3].

3.1. RFR-CNN

Two models we submitted are RFR-CNNs, which are based on CP *ResNet* [8, 2]. In the last DCASE challenge RFR-CNNs were used as student models [2] and have performed particularly well. Table 1 shows a summary of our first model.

WIDTH	BLOCK	CONFIG
$1 \rightarrow 24$	Input	5×5, HS, P
24	R	3×3, ReLU, 1×1, P_f
24	R	3×3, ReLU, 3×3, P_f
$24 \rightarrow 48$	R	3×3, ReLU, 1×1
48	R	1×1, ReLU, 1×1
$48 \rightarrow 72$	R	1×1, ReLU, 1×1
72	R	1×1, ReLU, 1×1
$72 \rightarrow 10$	Classifier	Global Mean Pooling

Table 1: RFR-CNN architecture based on [2]. The R blocks are residuals, where the input is added to the output element-wise. 3×3 and 1×1 are different kernel sizes. HS stands for the hard swish nonlinearity. P stands for a 2×2 max pooling and P_f for a 2×1 max pooling over the frequency dimension respectively.

To comply with the task regulations, we have adjusted the model from [2] in two ways:

First, we reduced the width throughout the network. This is due to minimizing the number of parameters and keeping the receptive field. Reducing the model complexity was necessary because we use the model in float16 precision, instead of quantizing it to int8.

Secondly, we have changed the nonlinearity in the first convolutional layer from ReLU to the so-called *hard swish* as proposed in [12]. With this swap of nonlinearities, we were able to reduce the initial set of filters to less than 32 while maintaining similar accuracy.

This model consists of 59 804 parameters represented by 16-bit floating point numbers and needs 14.687 million multiply-accumulate operations (MACs).

For our second model, we have implemented additional optimizations. We have substituted the 3×3 convolutions from the second and third R-Block with *depthwise separable convolution* to achieve this. This technique, used in Howard et al. [13], has helped us get the model complexity down to 43 580 parameters, and 10.819 million MACs.

3.2. S4 model

The structured state-space sequence model (S4) was introduced by Gu et al. [3] as a new sequence-to-sequence model for modeling long-range dependencies. The model is based on the fundamental state space model (SMM) $x'(t) = Ax(t) + Bu(t), y(t) = Cx(t) + Du(t)$, and is initialized with a special state space matrix A which enables the model as a viable option for sequence modeling. Further development of the model also introduces an algorithm to unroll the recurrence of the model over time, resulting in a kernel. Therefore the model can also be seen as a convolution with a global kernel over the sequence length, resulting in efficient computation both in computational and memory complexity. For a more in-depth explanation of the model, we would like to refer the reader to the original paper [3]. One motivation for adapting this model was the promising results on a subset of the Speech Command dataset [14] with an accuracy of 93.96% [3]. Our first experiments on the TAU Urban Acoustic Scenes 2022 Mobile development dataset [6] confirmed that the model has the capacity to fit the training data, leading to good training loss but strong over-fitting.

Firstly, a CNN layer extracts 16x16 patches of the Mel-spectrogram with an overlap of 12 and projects it to the feature space of the sequence, similar to how it is done in Vision Transformers [15]. This is followed by a stack of S4 blocks, global mean pooling over the sequence length, and a linear projection to the class predictions. The original S4 block consists of a Layernorm, an S4 layer, a Linear layer, and a skip connection between input and output. With the goal to reduce over-fitting and as well as shrink the model size, we propose the following changes to the model:

- We adapted Liquid-S4, a variant of the original S4 model, for better generalization [16]. This variant introduces an input-dependent state transition model, which makes it adaptive during inference.
- Similar to MobileNets [17], we swapped the Linear layer in the S4 block, with a point-wise convolution. Since the S4 layer can be seen as a depth-wise separable convolution, the idea and effects are very similar as in MobileNets.
- For better generalization, we introduced Mixup with $\alpha = 0.3$ for data augmentation. We also experimented with Freq-MixStyle, but Mixup worked the best.
- We also fine-tuned different hyperparameters, especially the δt _max and _min, as well as l_max ,which controls the effective size of the kernel [3], had a high impact.
- We also switched the initialization strategy of A from ”LegS”, which is suited for modeling long dependencies to ”FouT”, which is better at memorization tasks [3].

Since training the model in half-precision decreased the accuracy by 8%, we keep the full-precision for both training and inference, increasing the model's memory complexity. For the bigger S4-1 model, we use a base channel size $W = 80$, and for the smaller S4-2 model, we use a base channel size of $W = 48$. The final model structure can be seen in Table 2

OUT CHANNELS	BLOCK	CONFIG
W	CNN	16x16, S12
W/2	S4	PC, $W \rightarrow W/2$
W/2	S4	PC, $W/2 \rightarrow W/2$
W/2	S4	PC, $W/2 \rightarrow W/2$
Global Mean Pooling		
Classifier $W \rightarrow 10$ Classes		

S12: stride 12
PC: point-wise convolution

Table 2: S4 architecture

	Baseline		**RFR-CNN-1**		**RFR-CNN-2**		**S4-1**		**S4-2**	
Scene label	Accuracy	Log loss	Accuracy	Log loss	Accuracy	Log loss	Accuracy	Log loss	Accuracy	Log loss
Airport	39.4	1.534	**43.7**	1.489	41.2	1.578	43.6	**1.334**	34.8	1.487
Bus	29.3	1.758	**67.6**	**0.982**	56.7	1.225	44.5	1.618	41.9	1.642
Metro	**47.9**	**1.382**	46.4	1.402	42.1	1.490	43.5	1.483	43.4	1.555
Metro station	36.0	1.672	49.5	1.441	**51.4**	**1.379**	35.7	1.814	27.5	1.963
Park	58.9	1.448	**80.9**	**0.724**	77.0	0.823	73.3	0.992	73.3	0.988
Public square	20.8	2.265	36.3	1.749	**37.0**	**1.731**	19.4	2.124	16.6	2.027
Shopping mall	51.4	1.385	51.7	1.356	58.7	1.231	61.0	1.113	**62.6**	**1.083**
Street, pedestrian	30.1	1.822	**37.4**	**1.698**	34.8	1.758	28.4	1.926	31.9	1.749
Street, traffic	70.6	1.025	**77.4**	**0.776**	75.8	0.840	74.0	0.933	75.7	0.893
Tram	44.6	1.462	**61.5**	1.183	60.4	**1.181**	43.2	1.620	43.5	1.705
Overall	42.9	1.575	**55.2**	**1.280**	53.5	1.323	46.7	1.496	45.1	1.509
MACs	29.23		14.69		10.82		0.57		**0.21**	
Model Size	46.512		119.608		87.160		116.008		**63.976**	

Table 3: Results for baseline and submitted models on the given evaluation dataset. Accuracy in %. MACs in million. Model size in KB.

4. EXPERIMENTS AND RESULTS

4.1. Training and test dataset

In all our experiments we used the DCASE 2023 challenge task 1 split for training/validation/testing. Each sample belongs to one of 12 European cities, 3 real (A, B, C) or 6 simulated devices (S1-S6), and to one of 10 different acoustic scenes, including ”airport”, ”bus”, ”metro”, ”metro station”, ”park”, ”public square”, ”shopping mall”, ”street pedestrian”, ”street traffic”, and ”tram”. The simulated devices S1-S6 are generated by using measured impulse responses and applying range compression to recordings of device A. Each sample has a length of one second.

4.2. Training details

For both RFR-CNNs, we apply the Adam optimizer with a learning rate schedule and train for 60 epochs. The learning rate is exponentially increased to 0.001 in a warmup phase of 5 epochs and after 19 epochs it is then linearly decreased to 1e-5 within 10 epochs. We use a weight-decay of 0.003 and a mini-batch size of 256. For the Freq-MixStyle we set $p = \alpha = 0.3$.

For the S4 models, we also apply the Adam optimizer with a learning rate schedule and train for 50 epochs. The learning rate is exponentially increased to 2e-4 in a warmup phase of 5 epochs and after 19 epochs it is then linearly decreased to 2e-5 within 10 epochs. We use a weight-decay of 0.0005 and a mini-batch size of 64. For the Mixup we initialize $\alpha = 0.3$ for the bigger model S4-1 and $\alpha = 0.2$ for the smaller S4-2.

We trained our models for the final submission on the whole development dataset. The energy consumption of training and inference are reported in table 4.

Model	**Training**	**Inference**
Baseline	2.806	0.045
RFR-CNN-1	2.964	0.048
RFR-CNN-2	2.431	0.047
S4-1	1.242	0.011
S4-2	1.136	0.010

Table 4: Energy consumption for submitted models in kWh.

4.3. Results

Table 3 presents the results of our experiments, which compare the performance of different models on the given evaluation dataset. The table includes accuracy and log loss for each scene label, as well as the overall accuracy and log loss for all scenes. Additionally, the table displays the number of Multiply-Accumulates (MACs) in million and the model size in KB for each model.

Overall, the RFR-CNN-1 model achieves the highest accuracy (55.2%) and the lowest log loss (1.280). Furthermore, the S4-2 model has the lowest model size (63.976 KB) and requires the fewest MACs (0.21 million). It is also worth noting that some models perform better for specific scene labels than others, even though the overall accuracy differs significantly. For instance, the Metro scene has the highest accuracy (47.9%) and the lowest log loss (1.382) with the Baseline model. The Park scene has the highest accuracy (80.9%) and the lowest log loss (0.724) with the RFR-CNN-1 model.

5. CONCLUSION

Our submission for the DCASE 2023 consists of two different approaches. The first approach is an adaption of the model used in the winning approach of DCASE 22 Task 1. We changed the ReLU activation in the first convolutional layer to *Hard swish* and decreased the model complexity by reducing the width of the network. With those settings we were able to beat the baseline accuracy by more than 10% with the drawback of a higher amount of 59804 parameters, resulting in 119.6 KB (16-bit float) model size and 14.687 MMACs, still below the DCASE task requirements. Additionally, we utilized depthwise separable convolution in our approach, resulting in a significant reduction in model complexity while maintaining a high level of accuracy. For the S4 approach, we introduced multiple regularization methods to deal with over-fitting. Due to the parameter efficiency and the rather low MAC count, we aimed to design a more efficient model compared to the baseline, while still beating its accuracy score. For future work, we could imagine different already established methods like Knowledge Distillation, Pruning, and Quantization to further lower the model complexity and improve accuracies.

6. ACKNOWLEDGEMENT

Participating in the course "Machine Learning and Audio: A Challenge" offered the four of us a great opportunity to become familiar with academic challenges. Throughout the course's duration, we found ourselves in an environment that supported and encouraged us in the process of experimenting with the task. For this reason, we want to give special thanks to Professor Gerhard Widmer's team at the Institute of Computational Perception at JKU Linz. In particular, we would like to mention Florian Schmid, Paul Primus, Shahed Masoudian, Khaled Koutini, and Tobias Morocutti.

7. REFERENCES

[1] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in dcase 2022 challenge," 2022. [Online]. Available: https://arxiv.org/abs/2206.03835

[2] F. Schmid, S. Masoudian, K. Koutini, and G. Widmer, "CP-JKU submission to dcase22: Distilling knowledge for low-complexity convolutional neural networks from a patchout audio transformer," DCASE2022 Challenge, Tech. Rep., June 2022.

[3] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," 2022.

[4] K. Zhou, Y. Yang, Y. Qiao, and T. Xiang, "Domain generalization with mixstyle," *CoRR*, 2021.

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

[6] 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://dcase.community/documents/workshop2018/proceedings/DCASE2018Workshop_Mesaros_8.pdf

[7] B. Kim, S. Yang, J. Kim, H. Park, J.-T. Lee, and S. Chang, "Towards robust domain generalization in 2d neural audio processing," 2021.

[8] K. Koutini, F. Henkel, H. Eghbal-Zadeh, and G. Widmer, "Low-complexity models for acoustic scene classification based on receptive field regularization and frequency damping," in *Proceedings of 5th the Workshop on Detection and Classification of Acoustic Scenes and Events 2020 (DCASE 2020), Tokyo, Japan (full virtual), November 2-4, 2020*, N. Ono, N. Harada, Y. Kawaguchi, A. Mesaros, K. Imoto, Y. Koizumi, and T. Komatsu, Eds., 2020, pp. 86–90.

[9] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *CoRR*, 2019.

[10] H. Eghbal-zadeh, K. Koutini, and G. Widmer, "Acoustic scene classification and audio tagging with receptive-field-regularized cnns," *Tech. Rep., DCASE 2019 Challenge*, 2019.

[11] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Receptive field regularization techniques for audio classification and tagging with deep convolutional neural networks," *CoRR*, vol. abs/2105.12395, 2021.

[12] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.

[13] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017.

[14] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[15] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[16] R. Hasani, M. Lechner, T.-H. Wang, M. Chahine, A. Amini, and D. Rus, "Liquid structural state-space models," 2022.

[17] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017.