CPJKU SUBMISSION TO DCASE21: CROSS-DEVICE AUDIO SCENE CLASSIFICATION WITH WIDE SPARSE FREQUENCY-DAMPED CNNS

Technical Report

Khaled Koutini[1], *Jan Schlüter*[1], *Gerhard Widmer*[1,2],

[1]Institute of Computational Perception (CP-JKU), [2]LIT Artificial Intelligence Lab,
Johannes Kepler University Linz, Austria
khaled.koutini@jku.at

ABSTRACT

We describe the CP-JKU team's submission for Task 1A *Low-Complexity Acoustic Scene Classification with Multiple Devices* [1] of the DCASE2021 Challenge. We use Receptive Field (RF) regularized Convolutional Neural Network (CNN) with Frequency Damping as a baseline. We investigate widening the convolutional layers while keeping the number of parameters low by grouping and pruning. We apply iterative magnitude pruning to sparsify the weights of the models. Additionally, we investigate an adversarial domain adaptation approach.[1]

Index Terms— acoustic scene classification, receptive field regularization, pruning

1. INTRODUCTION

Convolutional Neural Networks (CNNs) have been dominating the field of acoustic scene classification [2]. In particular, restricting the Receptive Field (RF) of CNNs has shown success in the DCASE'20 challenge [3, 4]. Also, ensembles of CNNs with millions of parameters achieved the best performance on this task in the presence of multiple recording devices [2–7].

Over-parameterizing neural networks can lead to better generalization [8]. This can be seen in different domains such as vision [9] and natural language processing [10]. In particular, Tan and Le [11] show that for CNNs, scaling up the depth and the width of the network can lead to better generalization on image classification tasks. However, Koutini et al. [12] showed that scaling up the depth without regularizing the receptive field can result in overfitting in acoustic scene classification tasks. On the other hand, scaling up the width of the network, i.e, increasing the number of channels in the convolutional layers, results in a quadratic increase in the number of parameters. Golubeva et al. [13] showed that increasing the width of CNNs with sparsification – to keep the same number of parameters of the model – can improve the performance, indicating that width plays an important role in generalization, even when we avoid over-parameterization.

Koutini et al. [14] compare different methods of model compression on acoustic scene classification. They show that pruning yields the best results.

[1]Source code available at : `https://github.com/kkoutini/cpjku_dcase21`

Table 1: Baseline Architecture

	CHANNELS	BLOCK	CONFIG	PARAMS
	W	INPUT	5×5 S=2	$25W$
$1\times$	W	R	$3 \times 3, 1 \times 1$, P	$10W^2/G$
$1\times$	W	R	$3 \times 3, 3 \times 3$, P	$18W^2/G$
$1\times$	W	R	$3 \times 3, 3 \times 3$	$18W^2/G$
$1\times$	W	R	$3 \times 3, 3 \times 3$, P	$18W^2/G$
		LINEAR	$W \to 2\dot{W}$	$2W^2$
	$2\dot{W}$	R	$3 \times 3, F \times F$	$36W^2/G+$ $4F^2W^2/G$
$K\times$	$2\dot{W}$	R	$1 \times 1, 1 \times 1$	$8W^2/G$
CLASSIFIER $2 \times W \to 10$ CLASSES GLOBAL MEAN POOLING				$20W$

P: 2×2 MAX POOLING.
R: RESIDUAL, THE INPUT IS ADDED TO THE OUTPUT

2. EXPERIMENTAL SETUP

We use an identical setup as the CP-JKU team's submission to DCASE 2019 [15], therefore, we refer the reader to the technical report [15] for details. We additionally use pitch-shifting by randomly changing the maximum frequency of the mel filter bank [16].

We trained our models on the whole development dataset [2], We did not use any additional external data.

3. ARCHITECTURES

We use the frequency-damped variant of CP_ResNet [17] as a baseline, since this variant have shown to generalize better in acoustic scene classification. We increase the width of the baseline and introduce grouping. We further decrease the depth by removing the additional 1×1 convolutional layers. Table 1 specifies the architecture configuration. We explain the hyper-parameters W, K, F in the following sections.

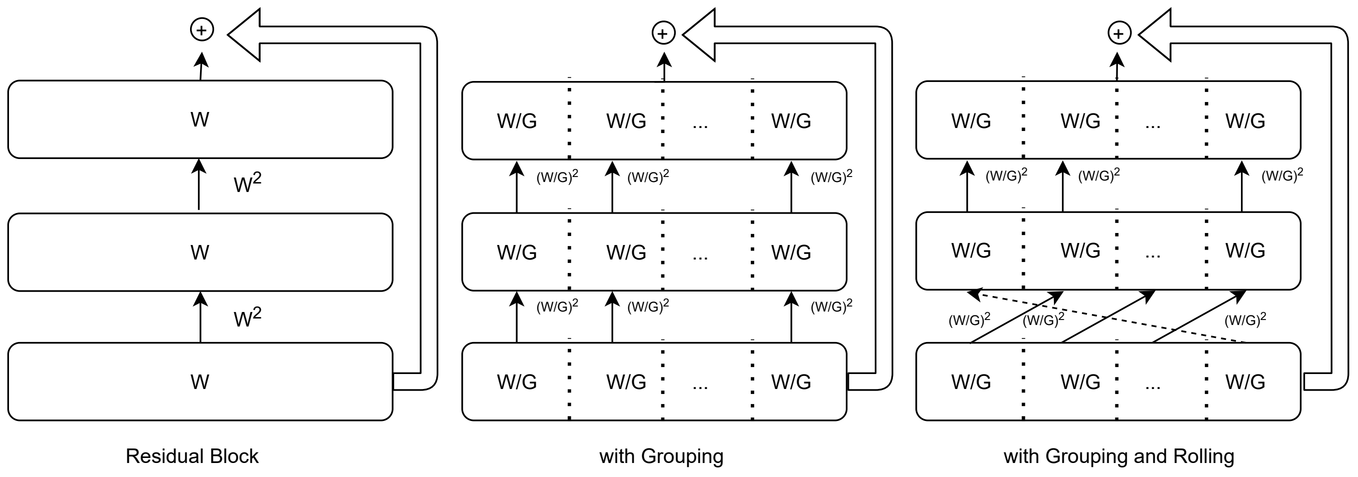

Figure 1: The effect of grouping on the residual blocks. Grouping over the channel dimension limits the connection to neurons within the group, reducing the number of connections (parameters) from W^2 to $G \times (W/G)^2 = W^2/G$. Rolling allows the information to be passed across groups.

3.1. Receptive Field and Depth

The experiments showed that the optimal Receptive Field (RF) for the Frequency-Damped network is achieved with $\rho = 7$, which corresponds to 135×135 pixels [17]. Tan and Lee [11] demonstrated that the performance of CNNs can also improve (on image classification tasks) by increasing the input resolution. Therefore, we also train models on spectrograms with larger frequency resolution, using 280 mel frequencies instead of 256. For these models, we compensate for the reduction of the relative RF with respect to the input size by setting $\rho = 8$, which corresponds to a RF of 167×167 pixels [17]. The networks with $\rho = 7$ and $\rho = 8$ correspond to $F = 1$ and $F = 3$ in Table 1, respectively.

In the original RF-regularized networks [17], there are many tailing 1×1 convolutional layers. These layers do not affect the receptive field of the network, and each residual block [17] (two convolutional layers) adds $8W^2/G$ parameters. Therefore, we remove these tailing layers, and experiment with $K = 0$ and $K = 1$ in Table 1.

3.2. Width

The width of the network can help improve the generalization capabilities of the model [11, 13]. However, increasing the width has a quadratic effect on the number of parameters of the network as shown in Table 1.

3.3. Grouping

We use grouped convolutional layers [18] as this has been shown to decrease the number of parameters with minimal impact on performance [17]. Grouping limits the connection of a convolutional layer to its input over the channels dimension, and therefore does not affect the RF of the network. A grouped convolutional layer with width W and G groups can be seen as G parallel convolutional layers of width W/G each. Using G groups reduces the number of parameters of a convolutional layer from $k \times k \times C_{in} \times C_{out}$ to $k \times k \times C_{in} \times C_{out}/G$, where k is the filter size, C_{in} is the input channels (width) and C_{out} is the output channels (see Figure 1).

In grouped convolutions, the output activation of each layer is affected only by the input within each group. Therefore, stacking more layers with the same number of groups results in keeping the information within each group throughout the network [19]. Therefore, we roll the input of each residual block over the channels dimension by the group size, connecting each group to its neighbouring group as shown in Figure 1. When combined with residual connections, this mixes information between groups.

3.4. Batch-Norm Layers

Batch-norm (BN) layers use the training data statistics to normalize their inputs during test time. Each BN layer in a CNN keeps a running average of the mean and variance of the training batches. Additionally, it introduces a trainable bias and scaling parameter for the layer's output. In CNNs, batch normalization is applied per channel. As a result, each BN layer stores $4 \times W$ parameters: $2 \times W$ trainable parameters (scale and bias) and $2 \times W$ non-trainable parameters, calculated from the training statistics (running mean and variance). The number $O(W)$ of these parameters is normally insignificant compared to the convolutional layers $O(W^2)$. However, we are using wide networks with $W = 256$, and since we do not prune these parameters in our pruning setup (as explained in Section 5), these parameter use a large portion of our $65K$ parameters limit: each layer adds $4 \times 256 = 1024$ parameters, adding up to $10K$ parameters in total. Therefore, we experiment with removing the scale and bias parameters (see Submissions 1, 3 in Section 6).

4. DOMAIN ADAPTATION

We use domain adversarial training in order to improve the generalization of the network on unseen devices. We use a residual block with two convolutional layers, parallel to the classifier block (Table 1) as a domain classifier. This layer operates on the embeddings of the network through a gradient reversal layer [20]. The Wasserstein distance is used to train the domain classifier with gradient penalty [21]. We use device A [2] samples as the source domain and all the other devices as a target domain.

ID	**W,G,K,ρ**	**DA**	**Size (KB)**	**Sparsity**	**Accuracy**				**Loss**			
					Overall	Real	Seen	Unseen	Overall	Real	Seen	Unseen
1	256,16,0,7	✗	127.78	87.02	68.57	73.60	70.26	61.87	0.92	0.76	0.90	1.09
2	256,16,1,8	✗	127.41	90.38	66.89	71.32	66.62	62.75	0.94	0.80	0.95	1.08
3	256,16,0,8	✗	127.67	89.71	69.46	75.33	68.80	64.27	0.89	0.72	0.91	1.04
4	256,16,0,8	✓	127.52	89.82	69.52	73.66	69.73	65.17	0.88	0.77	0.89	0.98
	128,1,1,7	✗	3869.79	0.00	70.70	73.34	71.71	67.04	0.88	0.77	0.87	1.01
	256,8,1,7	✗	3099.54	0.00	72.08	77.87	72.76	65.62	0.83	0.64	0.84	1.00
	384,32,1,7	✗	3959.29	0.00	72.37	78.09	72.53	66.48	0.83	0.66	0.85	0.98

Table 2: Comparison of different models on the Development Set. We report the accuracy and the loss over all classes. We report the mean of these metrics over the last 10 epochs of training. Note that we use half-precision (floating point 16bits) for the calculation of the model size. All models use frequency-damped convolutional layers. As explained in Table 1, **W** is the initial width of the network, **G** is the number of groups, **K** indicates the number of additional 1×1 convolutional layers, $\boldsymbol{\rho}$ controls the maximum RF [17]. **DA**: Domain Adaptation.

5. SPARSIFICATION

We decrease the depth of the network and apply grouping to the convolutional layers to reduce the network's size to around $500K$ parameters. We sparsify the network by zeroing out weights in order to reach the $65K$ parameters limit, in a similar fashion to [4].

We use iterative magnitude pruning [22] throughout training, since this has been shown to perform better than more recent pruning at initialization [23] approaches, such as SynFlow [24] and SNIP [25]. The number of weights removed in each epoch decays exponentially until we reach the target number $65K$ of parameters after 200 epochs of training. As a result, most of the weights are removed in the beginning of training.

We prune only the weights of convolutional layers, excluding the batch-norm weights from pruning. We also exclude the input layer and the classification layers from pruning, since they have a smaller number of parameters compared to the residual blocks (see Table 1), but have been found empirically to have a larger impact on performance when pruned. We use global unstructured pruning, by sorting all the weights globally (from all layers) by their magnitude and selecting the lowest to be zeroed out in each epoch. This method has shown to be superior to layer-wise and structured pruning.

We prevent the convolutional layers from collapsing [24] by excluding layers with more than 99% sparsity from further pruning. This simple heuristic turned out to be crucial in pruning wider networks.

6. SUBMISSIONS

Submission 1 (DampedR7NB) : We use the Frequency-Damped CP_ResNet $\rho = 7$ trained on spectrograms with 256 mel-frequencies, with width $W = 256$, grouped with rolling $G = 16$ and without any tailing 1 blocks $K = 0$. We also remove the scaling and the bias parameters of the batch-norm layers. The network has $504,104$ parameters. We sparsify it to have $64,690$ parameters of 16 bit float point, resulting in 126.35 KB and a sparsity of 87.17%.
Submission 2 (DampedR8) : We use the Frequency-Damped CP_ResNet $\rho = 8$ trained on spectrograms with 280 mel-frequencies, with width $W = 256$, grouped with rolling $G = 16$ and with one tailing 1 block $K = 1$. All the batch-norm layers have the scaling and the bias parameters in this submission. The network has $678,184$ parameters. We sparsify it to have $64,928$ parameters of 16 bit float point, resulting in 126.81 KB and a sparsity of 90.43%. We average the weights of the models in the last 10 epochs of training, Stochastic Weight Averaging (SWA) [26].
Submission 3 (DampedR8NB) : We use the Frequency-Damped CP_ResNet $\rho = 8$ trained on spectrograms with 280 mel-frequencies, with width $W = 256$, grouped with rolling $G = 16$ and without any tailing 1 blocks $K = 0$. We also remove the scaling and the bias parameters of the batch-norm layers. The network has $635,176$ parameters. We sparsify it to have $64,625$ parameters of 16 bit float point, resulting in 126.22 KB and a sparsity of 89.83%. This submission is similar to Submission 1, but with higher input resolution and larger $\rho = 8$. We average the weights of the models in the last 10 epochs of training (SWA).
Submission 4 (DampedR8DA) : is similar to Submission 3 $\rho = 8$, $W = 256$, $G = 16$ and $K = 0$ on inputs with 280 mel-frequencies. However, we keep the scaling and the bias parameters of batch-norm layers. We additionally use adversarial domain adaptation during training (as explained in Section 4). The resulting network has $641,320$ parameters. We sparsify it to have $63,529$ parameters of 16 bit float point, resulting in 124.08 KB and a sparsity of 90.09%.

7. RESULTS

Table 2 shows the results of different configurations on the development set. The rows numbered 1-4 represent models trained with the same configuration as the models submitted to challenge, trained on the development set[2]. The second part of the table shows the effect of widening the networks which improves the performance, even with small change to the number of parameters.

8. CONCLUSION

In this technical report, we described our submission to Task 1A of the DCASE-2021 challenge. We investigated scaling the width of CP_ResNet with Frequency Damping while keeping a low number of parameters by grouping and rolling. We use iterative pruning with an anti-layer-collapse heuristic to further sparsify the network and reduce the number of parameters of the model. We also investigate adversarial domain adaptation to help generalization to new devices.

[2]There are small differences in the final models' sizes and sparsity between the submitted models and the corresponding models on the development set. That's because pruning is a random process and can result in a slightly different final number of parameters.

9. 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. The LIT AI Lab is financed by the Federal State of Upper Austria.

10. REFERENCES

[1] I. Martín-Morató, T. Heittola, A. Mesaros, and T. Virtanen, "Low-complexity acoustic scene classification for multi-device audio: analysis of dcase 2021 challenge systems," 2021.

[2] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in DCASE 2020 Challenge: generalization across devices and low complexity solutions," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, 2020. [Online]. Available: https://arxiv.org/abs/2005.14623

[3] S. Suh, S. Park, Y. Jeong, and T. Lee, "Designing Acoustic Scene Classification Models with CNN Variants," DCASE2020 Challenge, Tech. Rep., 2020.

[4] K. Koutini, F. Henkel, H. Eghbal-zadeh, and G. Widmer, "CP-JKU Submissions to DCASE'20: Low-Complexity Cross-Device Acoustic Scene Classification with RF-Regularized CNNs," DCASE2020 Challenge, Tech. Rep., 2020.

[5] H. Hu, C.-H. H. Yang, X. Xia, X. Bai, X. Tang, Y. Wang, S. Niu, L. Chai, J. Li, H. Zhu, F. Bao, Y. Zhao, S. M. Siniscalchi, Y. Wang, J. Du, and C.-H. Lee, "Device-Robust Acoustic Scene Classification Based on Two-Stage Categorization and Data Augmentation," DCASE2020 Challenge, Tech. Rep., June 2020.

[6] W. Gao and M. McDonnell, "Acoustic Scene Classification Using Deep Residual Networks with Focal Loss and Mild Domain Adaptation," DCASE2020 Challenge, Tech. Rep., June 2020.

[7] L. Jie, "Acoustic Scene Classification with Residual Networks and Attention Mechanism," DCASE2020 Challenge, Tech. Rep., June 2020.

[8] M. Belkin, D. Hsu, S. Ma, and S. Mandal, "Reconciling modern machine-learning practice and the classical bias–variance trade-off," *Proceedings of the National Academy of Sciences*, vol. 116, no. 32, pp. 15 849–15 854, 2019. [Online]. Available: https://www.pnas.org/content/116/32/15849

[9] B. Neyshabur, Z. Li, S. Bhojanapalli, Y. LeCun, and N. Srebro, "The role of over-parametrization in generalization of neural networks," in *International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019. [Online]. Available: https://arxiv.org/abs/1805.12076

[10] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *CoRR*, vol. abs/2001.08361, 2020.

[11] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the International Conference on Machine Learning*, 2019, pp. 6105–6114. [Online]. Available: https://arxiv.org/abs/1905.11946

[12] 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. [Online]. Available: https://arxiv.org/abs/1907.01803

[13] A. Golubeva, B. Neyshabur, and G. Gur-Ari, "Are wider nets better given the same number of parameters?" in *International Conference on Learning Representations, ICLR 2021*, 2021. [Online]. Available: https://arxiv.org/abs/2010.14495

[14] 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 the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, Tokyo, Japan, November 2020, pp. 86–90.

[15] 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," DCASE2019 Challenge, Tech. Rep., June 2019.

[16] J. Schlüter, "Learning to monitor birdcalls from weakly-labeled focused recordings," in *Proceedings of the Twelfth International Conference of the CLEF Association (CLEF 2021)*, 2021.

[17] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Receptive field regularization techniques for audio classification and tagging with deep convolutional neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1987–2000, 2021.

[18] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *IEEE Conference On Computer Vision and Pattern Recognition*, 2017, pp. 1492–1500.

[19] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, 2018, pp. 6848–6856.

[20] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. S. Lempitsky, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, pp. 59:1–59:35, 2016.

[21] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 5767–5777.

[22] S. Janowsky, "Pruning versus clipping in neural networks." *Physical review. A, Atomic, molecular, and optical physics*, vol. 39, no. 12, pp. 6600–6603, 1989.

[23] J. Frankle, G. K. Dziugaite, D. M. Roy, and M. Carbin, "Pruning neural networks at initialization: Why are we missing the mark?" in *International Conference on Learning Representations, ICLR 2021*, 2020. [Online]. Available: https://arxiv.org/abs/2009.08576

[24] H. Tanaka, D. Kunin, D. L. Yamins, and S. Ganguli, “Pruning neural networks without any data by iteratively conserving synaptic flow,” in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. [Online]. Available: https://arxiv.org/abs/2006.05467

[25] N. Lee, T. Ajanthan, and P. H. S. Torr, “Snip: single-shot network pruning based on connection sensitivity,” in *International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019. [Online]. Available: https://openreview.net/forum?id=B1VZqjAcYX

[26] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson, “Averaging weights leads to wider optima and better generalization,” *arXiv preprint arXiv:1803.05407*, 2018.