Improving Scene Classification Models for Audio Coding Artifacts

Nils Peters

Friedrich-Alexander-Universität Erlangen-Nürnberg, International Audio Laboratories Erlangen[*]*, Germany*

Email: nils.peters@fau.de

Introduction

Acoustic Scene Classification (ASC) is the machine listening task to associate a semantic label from an audio recording that identifies the environment in which it has been captured [1]. Due to steady improvements in this area, machine listening approaches can now achieve ASC accuracy above human abilities [2]. However, ASC accuracy can significantly degrade when the classified audio signal is compressed by popular perceptual audio codecs, which were developed for human rather than machine listening. Audio compression likely occurs when the audio signals are efficiently stored prior ASC, or streamed, e.g., in an IoT scenario, where the ASC is performed as a downstream task on a cloud server. Rao and Peters reported a degradation by up to 57% compared to uncompressed audio signals (depending on codec and bit rate) and that a retraining of the classification model using previously compressed audio signals increases the accuracy when classifying perceptually coded signals. While this approach yields significant improvements, it may not be practical due to the required time and cost for complete retraining. Rather than a complete model retraining, this contribution investigates the option of fine-tuning a model pre-trained on uncompressed signals.

Strategies for ASC Model Improvements

To improve robustness of ASC against various audio coding artifacts, different approaches may be feasible:

1. As demonstrated in [3], retraining a model architecture with compressed audio signals can significantly improve model accuracy even when the inferred signals have been perceptually coded with codec configurations not used during retraining. Besides the very time-consuming efforts of retraining, the complexity of the inference does not increase.

2. It may be impractical (or impossible) to retrain the complete model. In such case, as a second option, one may use a post filter for blind signal enhancement of the audio signals prior to acoustic scene classification. For instance, a DNN-based post-filter was proposed in [4] to improve the perceived quality of highly compressed speech signals. In this case, the original ASC model is preserved, but additional run-time complexity caused by the post-filter enhancement prior inference is needed.

3. In the *fine-tuning* strategy, the training of an existing (pre-trained) ASC model continues for a number of epochs, using original and coded signals. This method is significantly faster than the retraining approach.

[*]The International Audio Laboratories are a joint institution between the Univ. of Erlangen-Nürnberg and Fraunhofer IIS.

The Baseline Model

To facilitate reproducibility, all experiments are based on the pre-trained acoustic scene classification model *10class-fcnn-model-0.7694* which was made available by the developers and authors of [5]. This pre-trained model is one component of a 5-part ensemble which was one of the best performing proponents of the DCASE 2020 Scene Classification Challenge [6], where 10 sec. recordings had to be classified to be one of 10 different acoustic environments (airport, bus, park, shopping mall, etc.).

The model is a VGG-like architecture with log-mel spectrogram feature input. It is comprised of ca. 11.8M parameters, consisting of 9 convolution layers followed by Batch Normalization, Dropout, and max-pooling layers (see segments 1-9 in Fig. 1). Channel attention is applied to the output channels of the last convolutional layer, followed by a global pooling layer. The final softmax layer returns the probabilities for all 10 acoustic scene classes.

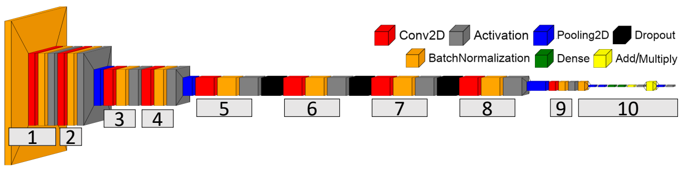

Figure 1: Visualization of the model architecture. Number boxes indicated the model segmentation for Experiment B

Experiment A: Full Model Fine-Tuning

In this experiment, the entire ASC model (i.e., all 11.8M parameters) is fine-tuned, using the dataset previously used for the retraining experiments in [3]. The dataset consists of audio files of the DCASE2020 training set [7], extended with versions compressed (see Table 1). The validation set consists of all audio signals of the official test split, compressed with different audio codecs. As visible in Table 1, three categories are created. Category 1 is comprised of the same codecs and bitrate settings which were also used for the fine-tuning. Category 2 is created from codecs that are used during fine-tuning, but

Table 1: Perceptual audio codecs and bitrates (kbps) used for fine-tuning

Codec	Fine-Tuning	Validation		
		cat 1	cat 2	cat 3
AAC (LC)	32	32	48, 64	
HE-AAC (v1)	16, 32	-	-	-
MP3	64	64	32	-
Opus	-	-	-	64
SBC	-	-	-	64

with different bitrate settings. The Category 3 consists of codecs not utilized for fine-tuning.

The model is fine-tuned over 40 epochs using stochastic gradient descent with a cosine-decay-restart learning rate scheduler. Table 2 shows the model accuracy at different epochs during the fine-tuning experiment. Without any fine-tuning (Epoch 0), the model performance significantly differs across evaluated codec conditions between 0.256 (MP3 at 32kbps) and 0.769 (WAV). With just a few fine-tuning epochs, the performance improves across conditions and peaks at 10 epochs, suggesting that fine-tuning beyond 10 epochs is not useful. Noteworthy, the accuracy of the uncoded condition (WAV) tends to continuously degrade slightly with the number of epochs. The accuracy improvements are comparable with those achieved with time-consuming retraining as in [3].

Table 2: Accuracy of the model for different epochs during fine-tuning. Subscript in the codec name indicates the bitrate in kbps.

Tuning Epochs		0	1	10	20	40
Codecs						
cat 1	WAV	**.7692**	.7520	.7379	.7382	.7204
	AAC_{32}	.5788	**.6958**	.6944	.6951	.6806
	$MP3_{64}$	.6745	.7069	**.7123**	.7018	.6951
cat 2	$MP3_{32}$	.2564	**.6398**	.6324	.6277	.5836
	AAC_{48}	.6412	.7001	**.7069**	.6988	.6873
	AAC_{64}	.6860	.7183	**.7146**	.7025	.6961
cat 3	$Opus_{64}$	**.6796**	.6496	.6715	.6691	.6557
	SBC_{64}	.5330	.6429	**.5916**	.5745	.5404

Experiment B: Segmental Fine-Tuning

In Experiment A, all 11.8M parameters were fine-tuned. To better understand which network layers are mostly sensitive (or important) to audio coding artifacts, in this experiment, the model is organized in 10 segments (as indicated in Fig. 1) and each segment is fine-tuned individually, whereas all other layers remained untouched. For each of these 10 experiments, the model is fine-tuned for 10 epochs. The results are depicted in Fig. 2 as relative improvements compared to the pre-trained model (i.e., no fine-tuning). For comparison, this Figure also shows the improvements for fine-tuning the full model (dashed lines). With respect to the segmental fine-tuning (solid lines), it is remarkable that even by fine-tuning a single segment, the performance can be significantly increased. Especially the late CNN layers (e.g., layer 7) can boost accuracy (e.g., for $MP3_{32}$ and AAC_{48}) and achieve performances similar or even better to those in Experiment A. This result is somewhat surprising because one could argue that fine-tuning may primarily change the input layer by discarding high-frequency features which are usually be affected by perceptual audio coding.

Summary and Conclusion

To make large ASC models more robust in the context of perceptual audio coding artifacts, this paper studied two fine-tuning methods. Compared to the previously proposed method of model retraining, it was shown that fine-tuning can increase model accuracy with much less training effort. Furthermore, it was shown that fine-tuning can be further optimized by fine-tuning of only small portions of the model, which makes this approach even more efficient. These findings are beneficial to better understand the inner workings of machine learning models and to improve ASC robustness whenever audio signals are subject to perceptual audio coding, e.g., due to transmission or lossy data storage.

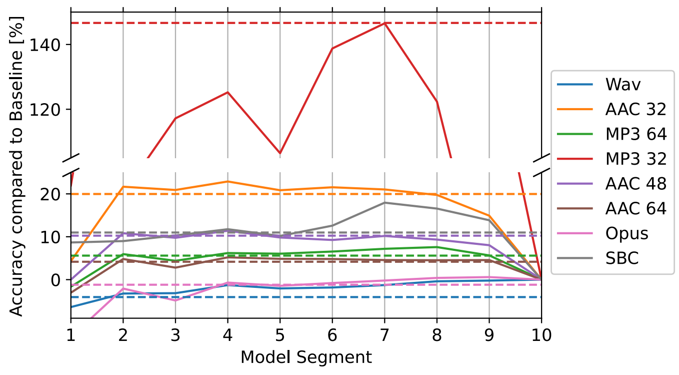

Figure 2: Accuracy improvements for fine-tuning single model segments. dashed lines: accuracy when fine-tuning full model as in Table 2, solid lines: accuracy from fine-tuning model segment

References

[1] Daniele Barchiesi et al., "Acoustic scene classification: Classifying environments from the sounds they produce," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 16–34, 2015.

[2] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen, "Assessment of human and machine performance in acoustic scene classification: DCASE 2016 case study," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017, pp. 319–323.

[3] Nagashree K. S. Rao and Nils Peters, "On the effect of coding artifacts on acoustic scene classification," in *Proc. of the 2021 Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2021.

[4] Srikanth Korse, Kishan Gupta, and Guillaume Fuchs, "Enhancement of coded speech using a mask-based post-filter," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6764–6768.

[5] Hu Hu et al., "Device-robust acoustic scene classification based on two-stage categorization and data augmentation," 2020, `https://arxiv.org/abs/2007.08389`, [Source Code:] `https://github.com/MihawkHu/DCASE2020_task1`.

[6] Toni Heittola, Annamaria Mesaros, and Tuomas Virtanen, "Acoustic scene classification in dcase 2020 challenge: Generalization across devices and low complexity solutions," in *Proc. of the Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, 2020.

[7] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen, "A multi-device dataset for urban acoustic scene classification," in *Proc. of the Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, 2018.