DOI: 10.61782/fa.2025.0940

FORUM ACUSTICUM EURONOISE 2025

URBAN SOUND CLASSIFICATION ON THE EDGE: EXPLORING THE ACCURACY-EFFICIENCY TRADE-OFF

Lion Cassens[1*] **Maarten Kroesen**[1] **Simeon Calvert**[2]
Sander van Cranenburgh[1]
[1] Department of Engineering Systems & Services, Delft University of Technology, the Netherlands
[2] Department of Transport & Planning, Delft University of Technology, the Netherlands

ABSTRACT

Sound source classification is a valuable addition to noise monitoring, providing `further insights into local soundscapes. For privacy preservation, this classification often must be conducted on the edge, i.e., in real time on noise sensors. This puts constraints on the size and complexity of the classification models that can be used. Furthermore, there is a trade-off between accuracy and efficiency, which needs to be balanced on battery or solar powered sensors. However, little is known about this trade-off under consideration of constraints imposed by such sensors. In this paper, we explore the scope of sound classification models that can run efficiently on low-cost sound sensors. Specifically, we investigate the Pareto frontiers between model accuracy and computational complexity, providing insights into the trade-off necessary for deploying such models on very constrained hardware. Building on these findings, we train new classification models optimized for edge devices. The models are trained on publicly available audio samples and a new Dutch Urban Sounds dataset specifically collected to enhance the accuracy of sound source classification in urban environments. The models and implementation are open source, enabling researchers and practitioners to adopt, adapt, and build upon our work.

[*]***Corresponding author***: *l.cassens@tudelft.nl.*

Keywords: *environmental sound classification, edge AI, convolutional neural network*

1. INTRODUCTION

Various studies have shown the importance of perception to explain the human response to the urban soundscape[1]. Annoyance, as well as recreational effects of the sound environment, depend not only on loudness but also on various other indicators. One important factor is the presence or absence of different sound sources. Bird song may be perceived by most as pleasant, while traffic noise is more likely to create a negative stress response. Machine learning models can be used to predict the presence of such sources. For privacy purposes, it is often necessary to run such models on a sensor, which requires small and efficient models.

While various papers propose such small models, few consider the end-to-end performance on edge devices (e.g., execution time including preprocessing), and practical constraints of microprocessors (such as limited for certain deep learning architectures).

In this paper, we define various models and analyze the trade-off between predictive performance and on-device efficiency. This trade-off is very important for battery or solar-powered sensors, as more efficient models result in lower energy consumption. Lastly, three Pareto optimal models along this trade-off are identified.

2. METHODS

The study follows the following steps: potential model candidates are created based on constraints imposed by microcontrollers. Then, these models are pretrained on public datasets and finetuned to a new dataset on Urban

Dutch Sounds. This dataset is then used to measure the predictive performance of all models. The on-device execution time of all models is measured as a proxy for energy efficiency. Lastly, Pareto optimal models are identified along the trade-off of predictive performance and efficiency.

2.1 Microcontroller constraints

Microcontrollers are energy and cost-efficient and, therefore, well-suited for low-cost or energy-constraint noise sensors. Yet, these low-memory devices impose several hardware and software limitations for machine learning models on the edge.

One difference compared to a desktop computer is the limited memory. Microcontrollers typically have less than 8MB RAM, which needs to be shared between sound source classification and all other tasks. Therefore, the model weights and any intermediate results and variables must not occupy more than a few MB of memory. Another difference is the limited processing speed, which is important because edge AI requires sound to be processed in real time. Thus, the classification time for a 3-second audio sample must not exceed 3 seconds.

Besides hardware constraints, there are further limitations in the support and optimization of certain ML operations. While it is theoretically possible to implement and optimize all required operations from scratch, this requires extensive knowledge and effort, which is often ignored in literature proposing very small ESC models. For this analysis, we constrain the choice of machine learning models to those that are supported and optimized for microcontrollers by TensorFlow Lite Micro. This excludes Transformer models and Long Short-Term Memory models.

Lastly, resampling audio is computationally expensive on most microcontrollers. We assume that the proposed models will run alongside a dB(A) calculation, which benefits greatly from a high sample rate. Therefore, all proposed models process audio sampled at 48 kHz.

2.2 Model candidates

Based on the constraints imposed by hardware and software, a total of nine convolutional neural network (CNN) candidates are evaluated. Specifically, three CNN architectures with varying numbers of parameters are combined with three Mel spectrogram sizes: 32, 64, and 104 Mels. All Mel spectrogram sizes are computed from short-time furrier transforms (STFT) with a 1024 sample window and hop rate. Table 1 lists the three CNN architectures.

Table 1. Layers and number of filters for all three model architectures. MaxPooling2D (2x2) follow each Conv2D but were omitted. Flatten and Dropout layers were also omitted from the table.

	Number of filters in Conv2D and units in dense layer		
Layer	**Small**	**Medium**	**Large**
Conv2D (3x3)	16	32	64
Conv2D (3x3)	16	48	64
Conv2D (3x3)	32	64	64
Conv2D (3x3)	32	64	96
Conv2D (3x3)	48	64	128
Dense	64	128	128
Dense (output)	11	11	11

2.3 Datasets

All models were pretrained on public datasets and then finetuned and evaluated on recently collected sound samples from the Netherlands.

To cover a variety of sounds, samples from the following datasets have been combined for the pretraining of all models: Google's AudioSet[2], ESC-50[3], UrbanSound8K[4], SONYC-UST[5] and Amsterdam Sounds Serval[6].

Some samples do not contain any of the selected sound classes but have been included to ensure that other audio sources are not mistaken for the sources considered here (i.e., to reduce false positives).

2.3.1 Dutch Urban Sounds

In addition to the existing datasets, a novel dataset has been collected to improve the prediction performance of sound source classification in an urban context. These 3-second samples have been collected in the Dutch cities Amsterdam and Leiden. Ideally, the microphone used for training data collection should be the same microphone that will be used in a noise sensor running the model. Otherwise, differences in microphones might bias the model predictions. This dataset has been recorded with a TDK ICS-434 MEMS microphone, which is well-suited for noise sensors.

Finetuning the candidate models to these samples is expected to improve real world accuracy, as publicly available datasets such as Google AudioSet can have poor data quality and do not necessarily reflect real world conditions. Other datasets such as SONY-UST are collected in a realistic setting, but specific to the American context.

Table 2. Audio datasets and occurrences of relevant sound classes per dataset.

	AudioSet	SONY-UST	Urban Sound 8K	ESC-50	Amsterdam Serval	Dutch Urban Sounds	Total
Samples	33,220	18,509	8,730	2,000	1,629	1,727	65,815
Vehicle	7,790	10,025	1,000	40	863	405	20,123
Honking	2,043	2,382	428	40	132	18	5,043
Aircraft	2,811	-	-	80	-	40	2,931
Siren	2,443	1,502	929	40	-	190	5,104
Human	10,310	7,254	1,000	40	634	585	19,823
Bark	843	1,114	999	40	-	61	3,057
Bird song	2,485	-	-	40	-	271	2,796
Church Bell	1,089	-	-	40	-	52	1,181
Music	6,886	1,627	1,000	-	-	133	9,646
Wind	2,322	-	-	40	-	457	2,819
Rain	1,145	-	-	40	-	104	1,289

Table 2 shows the number of samples per sound class for each dataset. AudioSet, SONY-UST, and Dutch Urban Sounds are multilabel datasets, meaning each sound class can occur more than once per sample.

2.4 Evaluation procedure

All models are pretrained on the mentioned public datasets. Then, each model is finetuned with 5-fold cross-validation on our Dutch Urban Sounds dataset. This dataset is imbalanced (i.e., some sound classes are sparser than others). Therefore, we evaluate the prediction performance using macro precision, recall, and F1. Micro metrics consider all samples equally, while macro metrics consider all sound classes equally. The latter is more suitable for this imbalanced dataset because the real-world class distribution likely differs from the dataset's distribution. To balance precision and recall, the macro F1 score is used as the final indicator of model performance.

An optimal model not only needs to be accurate but also computationally efficient. Slower models with longer execution times require stronger hardware and lead to an increased energy consumption. Therefore, an optimal model is always a trade-off between prediction performance (e.g., accuracy or F1 score) and efficiency. We measure efficiency as the total time to run the prediction pipeline on the sensor (execution time). The pipeline contains preprocessing and the ML inference itself. The on-device execution time is also a good proxy for energy consumption.

Measuring both prediction performance and efficiency allows us to find models that are Pareto optimal. A model is considered Pareto optimal if no other model performs better in one metric (e.g., F1 score or efficiency) without performing worse in the other, meaning it offers an optimal trade-off between the two.

3. RESULTS AND DISCUSSION

Table 3 lists macro precision, recall and F1 scores over all classes. As discussed, the macro F1 scores are used to determine the best predicting model.

The medium and large 104 Mel models achieve the same macro F1 scores. This may be caused by the imbalanced and relatively small finetuning dataset, which may lead to overfitting of large models. The worst model based on macro F1 is the small 32 Mel model.

Table 3. Macro precision, recall, and F1 scores for all model candidates.

Model		Macro		
Architecture	**Mels**	**Precision**	**Recall**	**F1**
Small	**32**	74.41%	52.38%	59.96%
Small	**64**	76.89%	55.80%	63.41%
Small	**104**	73.34%	54.30%	61.38%
Medium	**32**	70.38%	55.82%	61.90%
Medium	**64**	73.30%	56.17%	62.52%
Medium	**104**	76.27%	58.46%	65.69%
Large	**32**	70.33%	55.99%	61.97%
Large	**64**	75.04%	58.23%	64.77%
Large	**104**	76.07%	59.05%	65.69%

Prediction performance alone, however, is not sufficient to identify an optimal model for resource constraint edge AI. Therefore, we now discuss the on-device speed of each model.

The execution time refers to the overall time required to make a sound source prediction on the sensor. In this case, time was measured on an ESP32-S3 running at 240 MHz. Continuous monitoring of soundscapes requires the execution time for processing of n-seconds audio to not surpass n-seconds. In this case, each audio sample is 3 seconds long. Table 4 shows that all but the Large 104 Mel models fulfill the real-time requirement. The table further shows that architecture size is more important than input size (which depends on the number of Mels) for the execution time of the candidate models. Preprocessing (creation of Mel spectrograms) poses a significant overhead for smaller models but less so for larger models. This is due to the computational expense of short-term furrier transforms (STFT), which is independent of the number of Mels. Only the conversion to Mels varies but is less computationally demanding than STFT.

Table 4. Execution time (including preprocessing and ML inference) for the model candidates.

Model		Execution time (ms)		
Architecture	**Mels**	**Preprocessing**	**Inference**	**Total**
Small	**32**	107	79	186
Small	**64**	113	155	268
Small	**104**	118	250	368
Medium	**32**	107	239	346
Medium	**64**	113	476	589
Medium	**104**	118	759	877
Large	**32**	107	1,040	1,147
Large	**64**	113	2,069	2,182
Large	**104**	118	3,844	3,962

After considering predictive performance and efficiency (represented by execution time) individually, we now discuss the trade-offs between both metrics.

The ideal trade-off depends on the specific use case and design requirements. Therefore, there is no universally best trade-off. Instead, we look at multiple Pareto optimal models along this trade-off, which form the Pareto front.

Figure 1 compares the total execution time and Macro F1 scores of all models. Three models are Pareto optimal, namely the small architecture with 32 Mels, the small architecture with 64 Mels, and the medium architecture with 104 Mels. The large architecture with 104 Mels does not fulfill the requirements of a Pareto optimal model because the medium architecture with 104 Mels has a better execution time for the same Macro F1 score. The remaining models are also inferior to at least one of the Pareto optimal models and can be rejected.

The three Pareto optimal models contain between 46,379 and 281,147 model weights and occupy between 58 and 296 KB of memory after quantization.

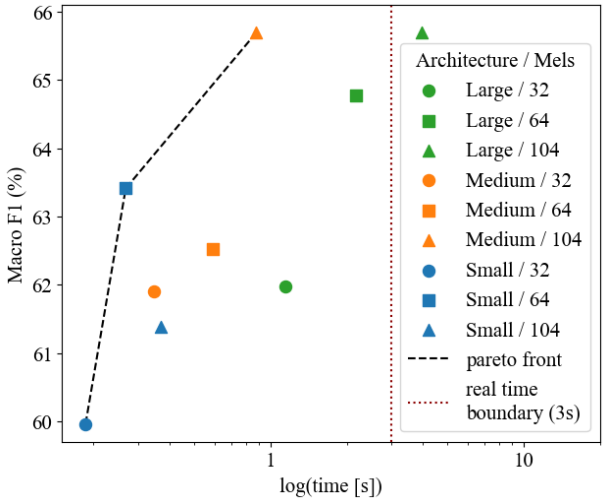

Figure 1. Total execution time and Macro F1 scores of all models and their Pareto front.

For training similar models on other datasets, it is recommended to repeat the Pareto analysis as the model candidates may perform differently depending on the dataset characteristics.

To provide a reference point for model performance, we compare the models with the state-of-the-art Audio Spectrogram Transformer[7] (AST) after also finetuning it on our Dutch Urban Sounds dataset. The AST reached a Macro F1 score of 74.87%. While this surpasses the best-performing model of this paper, the AST is also significantly larger (86,197,259 parameters). The finetuned AST is more than 300 times bigger than our largest Pareto optimal model and unsuitable for microprocessors.

Further research is necessary to compare our Pareto optimal models with other approaches, such as a training pipeline proposed by Mohaimenuzzaman[8].

4. CONCLUSION

Microprocessors impose various practical constraints on the choice of machine learning models due to hardware limitations or missing support and optimization for various deep learning architectures. This currently prohibits the use of Transformer and LSTM models with ML libraries such as TensorFlow Lite Micro.

Therefore, nine convolutional neural networks were compared regarding the trade-off between model performance (measured via Macro F1 scores) and efficiency (measured as total execution time). The most complex model did not fulfill real-time requirements but was also not found to be optimal. Instead, three Pareto-optimal models have been identified, each representing a different optimal trade-off between F1 score and efficiency.

The analysis shows the importance of considering efficiency in addition to model performance, as all six non-optimal models would lead to either more computational cost or less accurate sound source predictions. All three Pareto optimal models are openly available on GitHub[1] and can be used on microcontrollers supported by TensorFlow Lite Micro.

5. ACKNOWLEDGEMENTS

The research was funded by the TU Delft AI Initiative. The sound sample collection in Amsterdam was supported by the Responsible Sensing Lab of the Amsterdam Institute for Advanced Metropolitan Solutions.

6. REFERENCES

[1] Alvarsson, J.J., Wiens, S., Nilsson, M.E.: Stress Recovery during Exposure to Nature Sound and Environmental Noise. International Journal of Environmental Research and Public Health. 7, 1036–1046, 2010.

[2] Gemmeke, J.F., Ellis, D.P.W., Freedman, D., Jansen, A., Lawrence, W., Moore, R.C., Plakal, M., Ritter, M.: Audio Set: An ontology and human-labeled dataset for audio events. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 776–780, 2017.

[3] Piczak, K.J.: ESC: Dataset for Environmental Sound Classification. In: Proceedings of the 23rd ACM international conference on Multimedia. pp. 1015–1018. Association for Computing Machinery, New York, NY, USA, 2015.

[4] Salamon, J., Jacoby, C., Bello, J.P.: A Dataset and Taxonomy for Urban Sound Research. In: Proceedings of the 22nd ACM international conference on Multimedia. pp. 1041–1044. ACM, Orlando Florida USA, 2014.

[5] Cartwright, M., Cramer, J., Mendez, A.E.M., Wang, Y., Wu, H.-H., Lostanlen, V., Fuentes, M., Dove, G., Mydlarz, C., Salamon, J., Nov, O., Bello, J.P.: SONYC-UST-V2: An Urban Sound Tagging Dataset with Spatiotemporal Context, http://arxiv.org/abs/2009.05188, 2020.

[6] Amsterdam Sounds Serval on GitLab, https://gitlab.waag.org/lodewijk/amsterdam-sounds-serval/-/tree/master

[7] Gong, Y., Chung, Y.-A., Glass, J.: AST: Audio Spectrogram Transformer, 2021.

[8] Mohaimenuzzaman, M., Bergmeir, C., West, I., Meyer, B.: Environmental Sound Classification on the Edge: A Pipeline for Deep Acoustic Networks on Extremely Resource-Constrained Devices. Pattern Recognition, 2023.

[1] https://github.com/lioff/fa-euronoise-2025-urban-sound-classification