Discover Applied Sciences

Research

Training environmental sound classification models for real-world deployment in edge devices

Manuel Goulão[1,3] · Lourenço Bandeira[2] · Bruno Martins[1] · Arlindo L. Oliveira[1]

Received: 30 November 2023 / Accepted: 11 March 2024
Published online: 26 March 2024

Abstract

The interest in smart city technologies has grown in recent years, and a major challenge is to develop methods that can extract useful information from data collected by sensors in the city. One possible scenario is the use of sound sensors to detect passing vehicles, sirens, and other sounds on the streets. However, classifying sounds in a street environment is a complex task due to various factors that can affect sound quality, such as weather, traffic volume, and microphone quality. This paper presents a deep learning model for multi-label sound classification that can be deployed in the real world on edge devices. We describe two key components, namely data collection and preparation, and the methodology to train the model including a pre-train using knowledge distillation. We benchmark our models on the ESC-50 dataset and show an accuracy of 85.4%, comparable to similar state-of-the-art models requiring significantly more computational resources. We also evaluated the model using data collected in the real world by early prototypes of luminaires integrating edge devices, with results showing that the approach works well for most vehicles but has significant limitations for the classes "person" and "bicycle". Given the difference between the benchmarking and the real-world results, we claim that the quality and quantity of public and private data for this type of task is the main limitation. Finally, all results show great benefits in pretraining the model using knowledge distillation.

Article Highlights

- We describe the challenges of compiling a dataset for real-world environmental audio classification.
- We show how we used knowledge distillation to pretrain a tiny version of the model that can be used in edge devices.
- We analyze the current limitations of our model, training process and mainly our training data when we apply the models to real-world data.

Keywords Environmental sound classification · Edge devices · Smart cities · Deep convolutional neural network

✉ Manuel Goulão, manuel.silva.goulao@tecnico.ulisboa.pt; Lourenço Bandeira, lourenco.bandeira@schreder.com; Bruno Martins, bruno.g.martins@tecnico.ulisboa.pt; Arlindo L. Oliveira, arlindo.oliveira@tecnico.ulisboa.pt | [1]INESC-ID / Instituto Superior Técnico, Lisboa, Portugal. [2]Schréder Hyperion, Oeiras, Portugal. [3]NeuralShift, Lisboa, Portugal.

1 Introduction

As the world is becoming more and more connected, the interest in the development of technologies that enable cities to be more intelligent and sustainable is growing [32]. Machine learning plays a crucial role in this development, by enabling systems to learn from data collected by the different sensors installed in the city [2].

In the last decade, deep learning models have become a popular choice for several tasks in computer vision and audio processing, such as object detection, image segmentation, or audio and image classification. These models are capable of learning complex features from data, and their ability to generalize to unseen data has been shown in several real-world applications. As the field of deep learning keeps pushing for larger and more capable models, the interest in leveraging the knowledge learned by these models to solve real-world problems has been growing. However, in many applications, these models need to be deployed in the field using edge devices such as smartphones or IoT devices, which are limited in terms of computational power and energy consumption. To address those challenges, we have today several model compression techniques that aim to reduce the number of operations, the complexity of the operations, and the memory required to run these models, while preserving their accuracy. Some of those techniques are model pruning, quantization, weight matrix low-rank factorization, and knowledge distillation.

The project where this work is framed aims to develop a technological urban infrastructure based on the street luminaires, which will serve as the backbone for the implementation of technologies for the smart city transition with a special focus on the issue of mobility. This work presents the part of the project concerned with developing an ML model capable of detecting the presence of certain objects using audio signals. This model should be capable of identifying various classes of objects, such as pedestrians, cyclists, and vehicles, using visual and audio signals while running on edge devices integrated within luminaires. Moreover, the task considered is a multi-label classification, which is of great value for some of the planned applications, such as traffic control. This means that our system will be able to handle cases where multiple objects of the same class are present in a single audio, which is very common. We first present the dataset used for the development of the model. Then, we present the model architecture, which is based on a small ResNet backbone that also supports object detection on visual inputs [1], in the context of the same project. To further improve the accuracy of the model and decrease the computational cost, we describe how we used knowledge distillation to pretrain the tiny version of the ResNet. Finally, we present the results obtained with the model and discuss the challenges and limitations when applying our model to real-world data.

2 Related work

Environmental audio classification has been a topic of extensive research in recent years, with several studies proposing novel approaches for this task. Most of the early work with deep learning used convolutional neural networks (CNNs) to approach the problem. One major distinction between the different approaches is the type of features used as input to the CNN. Some of the most common features being used are raw signals [20, 28], log mel-spectrograms [11], log spectrograms [9, 15], gammatone frequency cepstral coefficients (GFCC), Constant Q-transform (CQT) features, Chromagrams, and a combination of those features [37, 45].

Since the audio signal is a time series, it is also possible to use recurrent neural networks (RNNs) to process it [41] or even the log mel-spectrograms of the audio [45]. More recently, several studies have explored the use of transformers instead of CNNs [6, 7, 14], which have achieved state-of-the-art results in several benchmarks. However, while some work has been done to optimize the computational cost of these models and make them suitable for edge devices [24, 26], these models are still not as fast as CNNs for many devices, which don't have dedicated operations for the transformer architecture. Moreover, given the more recent nature of these models, the number of published works using them is still relatively small compared to CNNs, which can make its implementation in a time constrained project more challenging.

To develop capable models with a high level of accuracy while keeping the computational cost low, several studies have explored model compression techniques and small architectures. AclNet [20] is a small CNN architecture which is capable of achieving state-of-the-art results on the ESC-50 dataset, by using waveform augmentations and mixup. Cui et al. [9] used knowledge distillation to transfer the knowledge learned from an ESResnet [15] to a modified MobileNetv2 [35] model. Mohaimenuzzaman et al. [28] proposed a network called ACDNet together with a model compression pipeline based on model pruning and quantization, that achieves almost state-of-the-art performance while being able to run on edge devices. The RACNN [11] uses a Resource Adaptive Convolutional (RAC) module to reduce the computational cost of the model, while maintaining classification accuracy.

In different areas of machine learning such as computer vision and natural language processing, the development of architectures designed to run on computationally limited devices has been a topic of extensive research. This is the case of several architectures such as MobileNet [19], YOLO [34], and SqueezeNet [21] among others. Moreover, several methods have been proposed to derive models capable of running in edge devices, by using model compression techniques such as pruning [16, 17, 44], quantization [16, 17], weight matrix decomposition [5], and knowledge distillation [4].

In this work, we take advantage of some of these techniques to put them to the test in a real-world scenario. Inspired by the work done by Palanisamy et al. [30] we explore reusing convolutional-based neural networks that were pretrained in computer vision tasks, which not only boosts the performance of the audio classification model but also reduces deployment time. Additionally, we employ knowledge distillation to transfer the knowledge learned from a large model SOTA model to our small model. Most importantly, we test these methods in a real-world deployment scenario, where we are able to demonstrate that the performance in popular benchmarks [33] doesn't translate to a real-world scenario.

3 Data

3.1 Data preparation

In this work we aimed to develop an audio classification model capable of detecting several classes of objects, including "Person," "Bus," "Siren," "Car," "Truck," "Motorcycle," and "Bicycle". The goal is to be able to detect the vast majority of objects that transit in a street environment and that were defined as relevant to the project. From the thousands of audio samples recorded from the luminaires, we labelled 1688, whereas the last 500 were collected and labelled later in the project. These newer samples had the advantage of being recorded synchronously with video which allowed us to improve the quality of the annotations by using not only the audio but also the video frames and respective predictions generated by a YOLO model. Nevertheless, the quantity and quality of the audio data produced by the early prototypes of the luminaires, also developed in the project, were limited. To address this limitation, we used a combination of data captured with the luminaires and publicly available datasets, including: (i) ESC-50 [33], i.e. a multi-class collection of environmental audio recordings with 50 classes from five categories, namely animals, natural soundscapes, human sounds, domestic sounds, and urban sounds; (ii) FSD50K [12], a dataset of audios comprising 200 classes drawn from the AudioSet Ontology; and (iii) the DCASE 2017 Task 4 strong label testing set [27], which is a dataset of audios extracted from Youtube videos comprising 16 classes usually found in a street environment and also drawn from the AudioSet Ontology. These datasets, except for ESC-50, were designed for multi-label classification tasks, which is also the case we are trying to solve with the audio model.

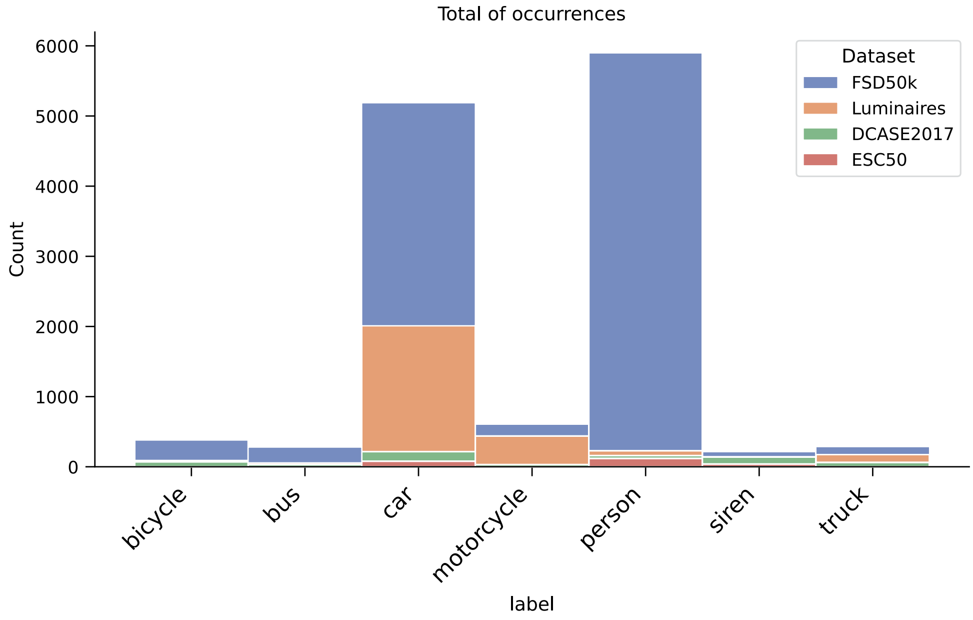
Compiling a dataset from these multiple sources presented several challenges. Firstly, some audio samples were not the expected 10 s long. Secondly, the different audio samples used different sample rates. Finally, the public datasets used different sets of classes. We solved the first two problems by normalizing the audio samples to 10 s, padding short audios and clipping long audios at the center, and resampling all audios to 22.05 kHz. For the third problem, we manually created a mapping between the classes present in the public datasets and the classes of interest for this project, see Tables 4 and 5.

Our final dataset consists of 12,961 audio samples with 36 h of sound in total. Table 1 shows the composition of the final dataset, and Fig. 1 shows the distribution of the audio samples per class in the final dataset.

Table 1 Composition of the final dataset

Source	Instances	Classes Present	Avg. Classes per Instance	Hours
Luminaires	1688	7	1.23	4.68
ESC-50	2000	3	1	0.66
FSD50K	8801	7	2.8	27
DCASE 2017	472	7	1.11	1.3

Fig. 1 Distribution of the audio samples per class in the final dataset



3.2 Audio visualization

Figure 2a shows the waveforms of three audio samples from three different datasets that contain the class “Motorcycle”. This figure shows, in part, the level of noise and the high level of amplitude of the signal that we have in the audio samples from the luminaires, especially when compared to the samples from the public datasets. In turn, Fig. 2b shows the mel spectrogram of the same waveforms. As we can observe, the spectrograms of the samples from the public datasets are more clear, with more well-defined high amplitude lines than the ones from the luminaires. Moreover, the spectrogram of the sample from the luminaires shows static noise across all frequency bins, with a higher prominence in the mel bins corresponding to the lower frequencies. This is confirmed by Fig. 2c, where we computed the median absolute deviation of the spectrogram across each frequency bin as an estimation of the noise level. We can observe that the noise level is higher in the samples from the luminaires, especially in lower frequencies. Furthermore, we also compared audios from early versions of the luminaires which contain a higher level of noise.

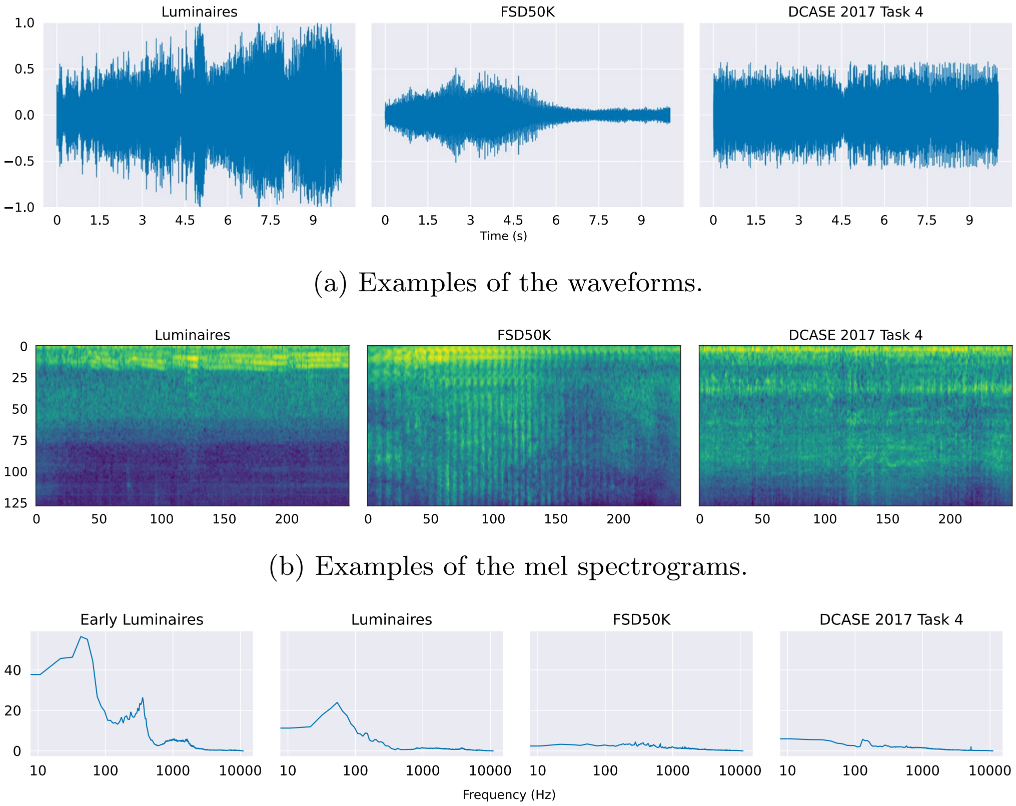
Even though additional work can be done to improve the quality of the data collected from the luminaires, we can see from the spectrograms, which will be used as input for the model, that the data collected by the luminaires is different from most of the audios we found in the public datasets. Nevertheless, given the limited amount of data from the luminaires and the poor balance of the class distribution, we still find these datasets of great value to train the audio model.

Additionally, while some of the limitations, such as noise, with the data collected with the luminaires might have been partially mitigated by noise cancellation techniques, we decided to keep the data as originally collected. The data was collected in real-world conditions and is therefore representative of the audio that the model will be exposed to when deployed.

4 Model

This section describes the different models we used for the multi-label audio classification task. All the models were developed and trained using PyTorch 1.10.1 [31], CUDA 11 [29], and Python 3.9 [39]. All our models were developed using a single NVIDIA V100S GPU and an Intel Xeon Silver 4214R CPU.

To make the development and training of the audio models more consistent with the environment used for the remaining models being developed for the project, we chose to write a model using PyTorch and reuse some of the existing codebases. Following previous work [30], which explores reusing CNNs developed for image classification



(a) Examples of the waveforms.

(b) Examples of the mel spectrograms.

(c) Median Absolute Deviation (MAD) of the Spectrogram across each frequency bin. We calculated the MAD for 50 different audios from each source and then averaged the results for each frequency bin.

Fig. 2 Visualizations of the audio samples from the different datasets

in an audio classification task we decided to adopt a ResNet backbone of an object detection model, which is a convolutional neural network with 3 channels as input and to which we added a classification block. We started with the YOLOv4-CSP [42] backbone and later changed it to the smaller and faster backbone of the YOLOv7-tiny [43] model. Although using CNN backbones for audio classification is common, most previous work considered inputs with one channel, while our backbones require three. To overcome this, we transform the 10 s long waveforms into three different mel spectrograms, each with 128 mel bands but with different sets of window sizes and hop sizes ((551, 220), (1102, 551), (2205, 1102)) for the short-time Fourier transform. The width of the spectrograms is normalized to 250, which we found to perform similarly to full-size spectrograms but with significantly faster computation. With this approach, we go from a 10 s waveform with 220500 dimensions that uses 4 bytes (float32) to a spectrogram with shape (3, 128,250) that also uses 4 bytes. Finally, we computed the log of the mel spectrogram values and normalized them using a mean of 4.5 and a standard deviation of 5.0.

We refer to the audio models as ResNet-tiny and ResNet, which are very similar with the only difference being the backbone used. Fig. 3 shows the architecture of the ResNet-tiny model with a classification block consisting of an adaptive average pooling operation with a targeted output of 1x1, a flattening operation, a linear layer with 1024 units, a hard swish activation, a dropout layer with a probability of 0.2, and a linear layer with the same number of units as the number of classes in the dataset. Given that we are solving a multi-label classification task we normalize the logits using the sigmoid function.

To use a model in an edge device, we exported it to the open standard ONNX and compiled it to TensorRT [8]. Our early observations showed no performance differences between the two versions, unfortunately, we were unable to perform more systematic evaluations which would allowed us to report a more confident result.

We compare the computational performance and memory usage of the models, as shown in Table 2. The ResNet-tiny model contains far fewer parameters, which results in a model that is significantly faster and uses less memory.

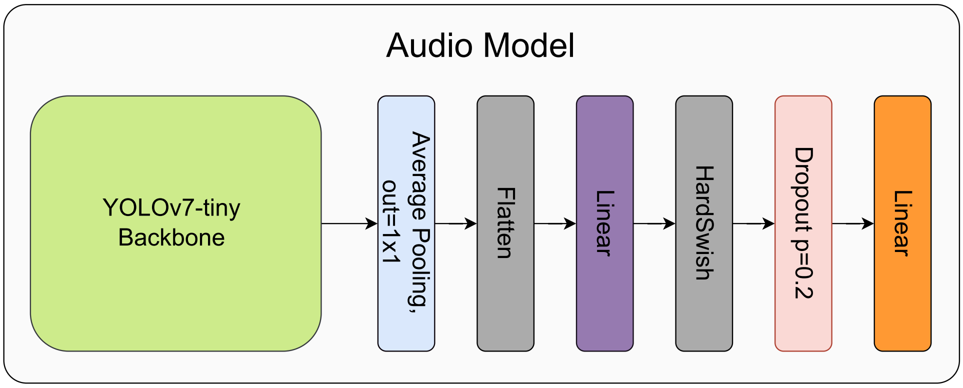


Fig. 3 Our ResNet-tiny audio classification model, which uses the backbone of a YOLOv7-tiny

Table 2 Comparison of computational performance and memory usage for the sound classification models. GFLOPS is the number of floating point operations per second given a spectrogram with shape (128, 256) which is the shape of the input during training and inference. Inference time is the time it takes to process a single spectrogram. Inferences per second are the number of spectrograms that can be processed per second. To measure it we used a batch size of 1 and 5000 batches. Memory usage is the sum of the memory used by the model parameters and buffers (not optimizable tensors, like batch normalization statistics)

Model	Parameters	GFLOPS	Inference time (ms)	Memory usage (MB)	Inferences per second
ResNet	26,610,535	6.24	12.8	78.0	106.6
ResNet-tiny	2,996,903	0.57	3.8	12.0	260.8

5 Pretraining with knowledge distillation

While the ResNet-tiny model is faster in comparison to the ResNet model, our experiments also showed a lower accuracy, which is expected given that it has a smaller size. To close the gap between the two, we pretrained the model with the ResNet-tiny backbone on the Audioset dataset [13] using knowledge distillation [18]. This approach allows us to compress the knowledge of a large model, or an ensemble of models, into a smaller model like the ResNet-tiny, and has shown to be effective in improving the performance of smaller models in previous work. We follow a procedure proposed in previous work [36], and use an ensemble of PaSST models [23] with different strides, that were pretrained on Audioset 2 M as the teacher model. More specifically, we use three PaSST models with strides 10, 12, and 14 and a patch size of 16. Given storage constraints, instead of using the complete Audioset 2 M dataset, for knowledge distillation, we only use the Audioset balanced training set and the evaluation set, which in total contain 33,679 audio samples.

We train the student model using two loss functions, namely the supervised loss (Eq. 2) and the knowledge distillation loss (Eq. 1), both corresponding to binary cross-entropy losses, where C is the number of classes. In one case one compares predictions against ground-truth labels, and in the other case, one compares predictions from the student the teacher models. The total loss is a weighted mean between the two, as shown in Eq. 3, where we give a higher weight to the knowledge distillation loss with $\alpha = 0.8$.

$$\mathcal{L}_{kd} = \frac{1}{C}\sum_{c}^{C} \sigma(T_c) \cdot \log \sigma(S_c) + (1 - \sigma(T_c)) \cdot \log (1 - \sigma(S_c)) \quad (1)$$

$$\mathcal{L}_{sup} = \frac{1}{C}\sum_{c}^{C} y_c \cdot \log \sigma(S_c) + (1 - y_c) \cdot \log (1 - \sigma(S_c)) \quad (2)$$

$$\mathcal{L} = (1 - \alpha)\mathcal{L}_{sup} + \alpha\mathcal{L}_{kd} \quad (3)$$

During the training loop, we augment the sound waveform with a random horizontal translation, also known as rolling. To compute the spectrograms, we use the procedure found in the source code of the PaSST model to extract the spectrograms for the teacher, and use the procedure presented in Sect. 5 for the student model. Both spectrograms are then separately augmented with time and frequency masking. Finally, before feeding the models with the spectrograms, we apply the same random mixup to the teacher and student spectrograms [3].

We pretrained the ResNet-tiny model using knowledge distillation for 10 epochs with a batch size of 64, mixup with $\lambda = 0.3$, and a learning rate with an exponential warmup for 3 epochs and a linear decay. To optimize the model, we use the Adam optimizer [22] with a weight decay of 0.0. We named the resulting model ResNet-tiny-distilled.

6 Training

We use the same training procedure to train the ResNet-tiny, ResNet-tiny-distilled, and ResNet models. Before training, we split the complete dataset into two folds using stratified sampling from each data source. At the start, the ResNet-tiny and ResNet models have a pretrained backbone that was pretrained using COCO 2017 [25], while ResNet-tiny-distilled is the model pretrained using knowledge distillation. All of them start with a classification block that is randomly initialized and with an output of 7 to account for the number of classes of interest. Our complete training/evaluation procedure consists of first training the model on the first fold and evaluating it on the second fold, and then training the model on the second fold and evaluating it on the first fold. Each training step consists of 300 epochs with a batch size of 64 and a learning rate of 1×10^{-5}. To optimize the model, we employ the binary cross-entropy loss and the Adam optimizer with a weight decay of 1×10^{-3}. Finally, we apply a set of augmentations to the spectrograms, namely rolling, time masking, and frequency masking. To avoid the cost of computing the spectrograms and considering that we need to train the model two times, we precompute the spectrograms and save them to disk. In the end, training the ResNet and ResNet-tiny models took a total of around 11 h and 30 min, and 4 h and 30 min, respectively.

7 Experimental results

Several metrics exist to evaluate the performance of models for a multi-label classification task. Some of those metrics are precision, recall, F1, and average precision. Precision is the ratio of true positives (TP) over the sum of true positives and false positives (FP). Recall is the ratio of true positives over the sum of true positives and false negatives (FN). F1 is the harmonic mean between precision and recall. Average precision is the area under the precision-recall curve, which is the curve that results from plotting the precision and recall values for different thresholds of confidence. Both F1 and average precision are metrics that balance precision and recall and for that reason are more commonly used. Considering that our task is multi-label classification, these metrics are computed per class and then averaged over all classes. To average the metrics, we can use the micro average or the macro average. The micro average is computed by summing the true positives, false positives, and false negatives over all classes, and then computing the metric. The macro average is computed by computing the metric for each class and then averaging the results.

For assessing the results of the audio models, we use the F1 score as the main metric and do a macro average over all classes. The results are the cross-validation scores of the models on the dataset previously presented, over 2 folds.

In Fig. 4, we can observe that the ResNet-tiny-distilled model is the best-performing in macro average F1 over all the classes, and for most of the classes. All models show a good performance for the classes that are more represented in

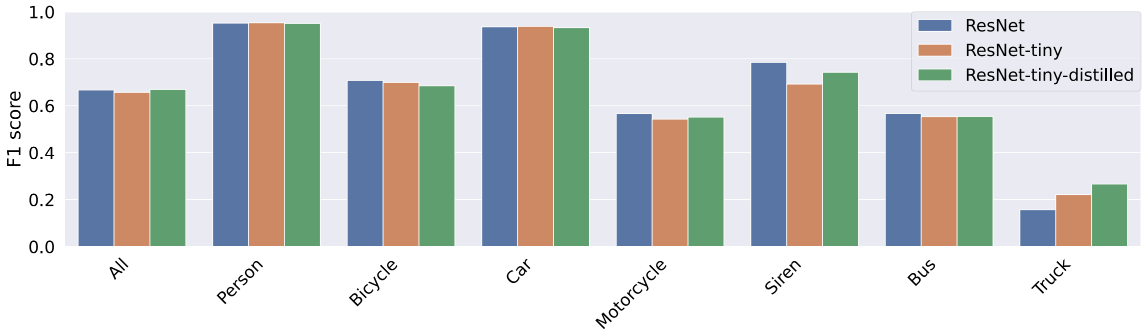


Fig. 4 Audio models F1 score per class and macro average for all classes

the dataset, such as "person" and "car". When we consider the classes that are less represented in the dataset, we observe cases where models show poor performance, such as "truck", "bus", and "motorcycle".

In Fig. 5, we show the same results but for the case of testing with audio samples recorded by the early prototypes of the luminaires. As we can observe the performance of the models is much lower than when testing with the full dataset. This shows the challenge of the task and in part the limitations of using audio to detect the presence of some classes of objects. However, the bad performance of the ResNet-tiny-distilled for some classes can be explained by the fact that those classes are extremely sub-represented in the data captured from the luminaires. For example, in the training dataset the classes "person", "bicycle", "siren and "bus" are only present in 69, 18, 1, and 24 audio samples, respectively, which are a very small fraction of the 1688 audios used in the dataset.

8 Benchmarking

To ground our results with previous work, we compare our model with the state-of-the-art models on the ESC-50 dataset [33]. ESC-50 is a dataset of 2000 audio samples and 50 classes of environmental sounds for multi-class classification, which is frequently used to benchmark audio classification models. To evaluate the models, the dataset is split into five folds and the evaluation metric is the accuracy of the cross-validation over the five folds.

Our training procedure is very similar to the one described in Sect. 7, and we did not perform any hyperparameter search to further improve the results. For each fold, we train the model for 500 epochs, with a batch size of 64, a starting learning rate of 1×10^{-4} that decreases 0.1 at every third of the training, and optimize the model using the cross-entropy loss and the Adam optimizer [22] with a weight decay of 1×10^{-3}. Furthermore, we use the same augmentation procedure with random horizontal translation, random frequency masking, and random temporal masking. However, we also use a random mixup with $\lambda = 0.2$. Unlike most of the models in the literature, we did not use a sample rate of 44.1kHz, instead resampled the instances to 22.05kHz, which is the sample rate we are using with the real-world data. Finally, to optimize the model, we use the cross-entropy loss and the Adam optimizer.

In Table 3 we compare our models against the state-of-the-art. As we can observe, our model ResNet-tiny-distilled achieves a performance comparable to some of the state-of-the-art models. Except for the ACDNet model, all the other alternatives have a much higher number of parameters and in some cases perform worse in terms of accuracy.

The most comparable model to our ResNet-tiny-distilled is the ACDNet model [28] which is a convolutional neural network designed for environmental sound classification on edge devices. The ACDNet achieves slightly better performance (+1.5%) while using a comparable number of parameters.

We find these results very interesting and promising, as they show that our model can achieve good performance in a standard audio classification task, even though its backbone was not designed for this type of task, like for example the ACDNet. Taking into account the results with real-world data in Sect. 8, we believe that most of the focus in future work should be on improving the data capture and data augmentation techniques.

9 Discussion and conclusions

This work presented an audio classification model developed to detect events in a street environment using sound.

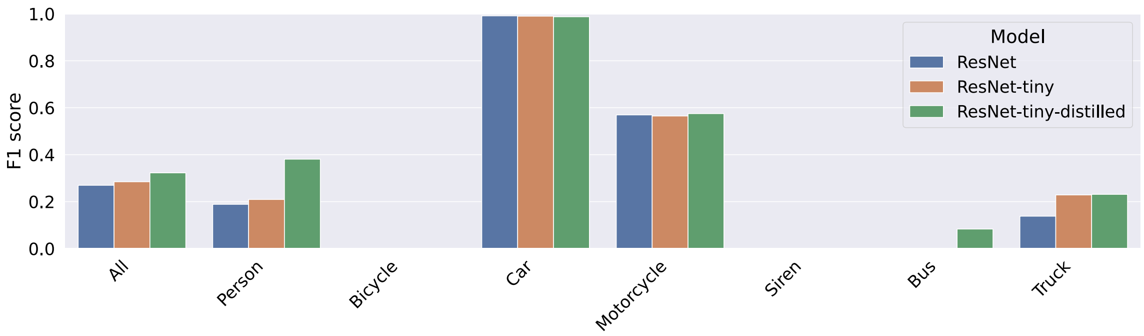


Fig. 5 F1 score per class and the macro average for all classes, when testing only with the samples audios recorded from the luminaires

Table 3 Comparison of the performance of different sound models on ESC-50

Model	# parameters	ESC-50 Accuracy
BEATs [7]	90M	97.45%
HTS-AT [6]	31M	97.00%
CLAP [10]	80.8M	96.70%
AST [14]	87M	95.70%
ACDNet [28]	4.74M	87.10%
AclNet [20]	10.63M	85.65%
ResNet-tiny-distilled **(ours)**	2.9M	85.4%
EnvNetv2 [38]	101M	84.90%
ResNet **(ours)**	26,6M	81.9%
ResNet-tiny **(ours)**	2.9M	81.6%

Our results show that a ResNet-tiny-distilled model, which uses as backbone a ResNet from an object detection model for visual inputs, is the best-performing model even when testing with audios recorded with the luminaires, and surpassing the performance of a larger ResNet model. However, the performance of the model is still fairly low when we evaluate it using real-world data. We think that there are two main reasons for this: the first one is the fact that the data collected with the luminaires is still very noisy, despite the effort to improve the quality of the audio. This is due to several factors like distance to the vehicles, wind, number of vehicles and other environmental noises, which in the end contribute to a much more complex signal as compared to the data from the public datasets, making it difficult for the model to generalize. The second reason is the class distribution unbalance, which is particularly evident in the data from the luminaires.

Future work will need to focus on improving the quantity and quality of the data by collecting more data and tuning the microphones in the luminaires. Nevertheless, our current results confirm that the task at hand is very challenging and that there are limitations in using audio to detect the presence of some classes of objects. For example, microphones do not reliably detect classes of objects that produce a low sound (e.g. a person or a bicycle). Also, as we note in Sect. 4.1, annotating the audio recorded from the luminaires is very difficult even for humans and the use of a synchronized video contributes immensely to the quality of the annotations. Despite the difficulty of detecting some classes, during the process of collecting and annotating more data we observed a continuous improvement in the performance of the models, which leads us to believe that we still have some performance to gain by improving the quantity and quality of the data. Additionally, classes like siren which are easy to detect in the audio recordings, are by contrast very rare in the real world, which makes the task of capturing audio with their presence challenging, affecting the performance of the models.

We hope that our work can serve as a starting point for future work in this area. The use of audio to detect events in a street environment is a promising area of research and a lot can be done to improve the performance of the models. While the existing public datasets are a good starting point, we show that the development of new datasets with data more representative of real-world conditions is crucial to obtaining capable models. Additionally, we hope that current and future efforts in this domain [40] can contribute to the construction of such datasets. Finally, as the field develops larger and more capable models we believe that improving model compression techniques like knowledge distillation is of great value.

Funding We acknowledge the financial support provided by Project Magellan, reference LISBOA-01-0247-FEDER-071980, the Recovery and Resilience Fund towards the Center for Responsible AI (Ref. C628696807-00454142), and the financing of the Foundation for Science and Technology (FCT) for INESCID (Ref. UIDB/50021/2020).

Data availability The data supporting the results presented in this work is not publicly available due to privacy concerns and ownership restrictions. The data generated during this research was recorded in a public street environment using early prototypes of luminaires with an edge device capable of recording audio. The installation of the luminaires and the recordings were authorized by the responsible state entity. As the nature of the data involves sensitive information, sharing openly is not feasible to comply with GDPR and legal obligations. The data is property of the Schréder Hyperion company, for further details and requests, please contact Lourenço Bandeira at lourenco.bandeira@schreder.com.

Declarations

Competing interests The authors declare no competing interests.

Class mapping for the audio datasets

See Tables 4 and 5 here

Table 4 Class mapping for the FSD50K dataset

Original	New
Bicycle	Bicycle
Bicycle_bell	Bicycle
Bus	Bus
Car	Car
Car_passing_by	Car
Chatter	Person
Child_speech_and_kid_speaking	Person
Conversation	Person
Cough	Person
Crowd	Person
Engine	Car
Engine_starting	Car
Female_singing	Person
Female_speech_and_woman_speaking	Person
Giggle	Person
Human_group_actions	Person
Human_voice	Person
Male_singing	Person
Male_speech_and_man_speaking	Person
Motor_vehicle_(road)	Car
Motorcycle	Motorcycle
Race_car_and_auto_racing	Car
Respiratory_sounds	Person
Run	Person
Screaming	Person
Shout	Person
Singing	Person
Siren	Siren
Skateboard	Bicycle
Sneeze	Person
Speech	Person
Traffic_noise_and_roadway_noise	Car
Truck	Truck
Vehicle	Car
Vehicle_horn_and_car_horn_and_honking	Car
Yell	Person

Table 5 Class mapping for the ESC-50 dataset

Original	New
Crying baby	Person
Siren	Siren
Footsteps	Person
Car horn	Car
Engine	Car
Laughing	Person

References

1. Arandjelović R, Zisserman A. Look, listen and learn. In: 2017 IEEE international conference on computer vision (ICCV); 2017. p. 609–617. https://api.semanticscholar.org/CorpusID:10769575
2. Atitallah SB, Driss M, Boulila W, Ghézala HHB. Leveraging deep learning and iot big data analytics to support the smart cities development: Review and future directions. Comput Sci Rev. 2020;38:100303.
3. Beyer L, Zhai X, Royer A, Markeeva L, Anil R, Kolesnikov A. Knowledge distillation: a good teacher is patient and consistent. In: 2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR);2022. p. 10915–10924.
4. Bharadhwaj M, Ramadurai G, Ravindran B. Detecting vehicles on the edge: knowledge distillation to improve performance in heterogeneous road traffic. In: 2022 IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW); 2022. p. 3191–3197.
5. Bhattacharya S, Lane ND. Sparsification and separation of deep learning layers for constrained resource inference on wearables. In: Proceedings of the 14th ACM conference on embedded network sensor systems CD-ROM. SenSys '16, association for computing machinery, New York, NY, USA; 2016. p. 176–189.
6. Chen K, Du X, Zhu B, Ma Z, Berg-Kirkpatrick T, Dubnov S. Hts-at: A hierarchical token-semantic audio transformer for sound classification and detection. ICASSP 2022 - 2022 IEEE international conference on acoustics, speech and signal processing (ICASSP); 2022. p. 646–650.
7. Chen S, Wu Y, Wang C, Liu S, Tompkins DC, Chen Z, Wei F. Beats: Audio pre-training with acoustic tokenizers; 2022. ArXiv abs/2212.09058
8. Corporation N. Nvidia tensorrt; 2023. https://developer.nvidia.com/tensorrt. Accessed on 24 Apr 2024.
9. Cui Q, Zhao K, Wang L, Gao K, Cao F, Wang X. Environmental sound classification based on knowledge distillation. In: 2022 16th IEEE international conference on signal processing (ICSP). vol. 1, IEEE; 2022. p. 245–249.
10. Elizalde B, Deshmukh S, Ismail MA, Wang H. Clap: Learning audio concepts from natural language supervision; 2022. ArXiv abs/2206.04769.
11. Fang Z, Yin B, Du Z, Huang X. Fast environmental sound classification based on resource adaptive convolutional neural network. Sci Rep. 2022;12:6599.
12. Fonseca E, Favory X, Pons J, Font F, Serra X. FSD50K: an open dataset of human-labeled sound events. IEEE/ACM Trans Audio Speech Lang Process. 2022;30:829–52.
13. Gemmeke JF, Ellis DPW, Freedman D, Jansen A, Lawrence W, Moore RC, 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); 2017. p. 776–780.
14. Gong Y, Chung YA, Glass JR. Ast: Audio spectrogram transformer. In: interspeech; 2021.
15. Guzhov A, Raue F, Hees J, Dengel AR. Esresnet: Environmental sound classification based on visual domain models. 2020 25th international conference on pattern recognition (ICPR); 2020. p. 4933–4940.
16. Han S, Kang J, Mao H, Hu Y, Li X, Li Y, Xie D, Luo H, Yao S, Wang Y, Yang H, Dally WJ. Ese: efficient speech recognition engine with sparse lstm on fpga. In: proceedings of the 2017 ACM/SIGDA international symposium on field-programmable gate arrays; 2016.
17. Han S, Mao H, Dally WJ. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. arXiv: computer vision and pattern recognition; 2015.
18. Hinton GE, Vinyals O, Dean J. Distilling the knowledge in a neural network; 2015. ArXiv abs/1503.02531
19. Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H. Mobilenets: efficient convolutional neural networks for mobile vision applications; 2017. ArXiv abs/1704.04861
20. Huang J, Leanos JJA. Aclnet: efficient end-to-end audio classification cnn; 2018. ArXiv abs/1811.06669
21. Iandola FN, Moskewicz MW, Ashraf K, Han S, Dally WJ, Keutzer K. Squeezenet: alexnet-level accuracy with 50x fewer parameters and < 1mb model size; 2016. ArXiv abs/1602.07360
22. Kingma DP, Ba J. Adam: a method for stochastic optimization; 2017.
23. Koutini K, Schlüter J, Eghbal-zadeh H, Widmer G. Efficient training of audio transformers with patchout; 2021. ArXiv abs/2110.05069
24. Li Y, Yuan G, Wen Y, Hu E, Evangelidis G, Tulyakov S, Wang Y, Ren J. Efficientformer: vision transformers at mobilenet speed; 2022. ArXiv abs/2206.01191, https://api.semanticscholar.org/CorpusID:249282517
25. Lin TY, Maire M, Belongie SJ, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL. Microsoft coco: common objects in context. In: European conference on computer vision; 2014.
26. Mehta S, Rastegari M. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer; 2021. ArXiv abs/2110.02178 https://api.semanticscholar.org/CorpusID:238354201
27. Mesaros A, Heittola T, Diment A, Elizalde B, Shah A, Vincent E, Raj B, Virtanen T. Dcase 2017 challenge setup: tasks, datasets and baseline system. In: DCASE 2017-workshop on detection and classification of acoustic scenes and events; 2017.
28. Mohaimenuzzaman M, Bergmeir C, West IT, Meyer B. Environmental sound classification on the edge: a pipeline for deep acoustic networks on extremely resource-constrained devices. Pattern Recognit. 2021;133:109025.
29. NVIDIA, Vingelmann P, Fitzek FH. Cuda, release: 10.2.89; 2020. https://developer.nvidia.com/cuda-toolkit

30. Palanisamy K, Singhania D, Yao A. Rethinking cnn models for audio classification; 2020.
31. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Köpf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S. Pytorch: an imperative style, high-performance deep learning library. In: neural information processing systems; 2019.
32. Perera C, Zaslavsky AB, Christen P, Georgakopoulos D. Sensing as a service model for smart cities supported by internet of things. Transactions on emerging telecommunications technologies 2013;25, https://api.semanticscholar.org/CorpusID:15340505
33. Piczak KJ. ESC: Dataset for Environmental Sound Classification. Proceedings of the 23rd ACM international conference on multimedia; 2015.
34. Redmon J, Divvala SK, Girshick RB, Farhadi A. You only look once: Unified, real-time object detection. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR); 2015. p. 779–788.
35. Sandler M, Howard AG, Zhu M, Zhmoginov A, Chen LC. Mobilenetv2: inverted residuals and linear bottlenecks. 2018 IEEE/CVF conference on computer vision and pattern recognition; 2018. p. 4510–4520.
36. Schmid F, Koutini K, Widmer G. Efficient large-scale audio tagging via transformer-to-cnn knowledge distillation; 2022. ArXiv abs/2211.04772
37. Sharma J, Granmo OC, Goodwin M. Environment sound classification using multiple feature channels and attention based deep convolutional neural network. In: interspeech; 2019.
38. Tokozume Y, Ushiku Y, Harada T. Learning from between-class examples for deep sound recognition; 2017. ArXiv abs/1711.10282
39. Van Rossum G, Drake FL. Python 3 Reference Manual. Scotts Valley, CA: CreateSpace; 2009.
40. Vryzas N, Stamatiadou ME, Vrysis L, Dimoulas CA. The beemate: air quality monitoring through crowdsourced audiovisual data. In: 2023 8th international conference on smart and sustainable technologies (SpliTech); 2023. p. 1–5. https://api.semanticscholar.org/CorpusID:260388048
41. Vu TH, Wang JC. Acoustic scene and event recognition using recurrent neural networks. Detect Classif Acoustic Scenes Events. 2016;2016:1–3.
42. Wang CY, Bochkovskiy A, Liao HYM. Scaled-yolov4: scaling cross stage partial network. In: 2021 IEEE/CVF conference on computer vision and pattern recognition (CVPR); 2020. p. 13024–13033.
43. Wang CY, Bochkovskiy A, Liao HYM. YOLOv7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors; 2022. ArXiv abs/2207.02696
44. Yao S, Zhao Y, Zhang A, Su L, Abdelzaher TF. Deepiot: Compressing deep neural network structures for sensing systems with a compressor-critic framework. Proceedings of the 15th ACM conference on embedded network sensor systems; 2017.
45. Zhang Y, Zeng J, Li YH, Chen D. Convolutional neural network-gated recurrent unit neural network with feature fusion for environmental sound classification. Autom Control Comput Sci. 2021;55:311–8.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.