Journal : SmallExtended 11227	Article No : 6796	Pages : 15	MS Code : 6796	Dispatch : 5-12-2024

The Journal of Supercomputing _####################_
https://doi.org/10.1007/s11227-024-06796-1

Edge computing for driving safety: evaluating deep learning models for cost-effective sound event detection

Carlos Castorena[1] · Jesus Lopez-Ballester[1] · Juan A. De Rus[1] · Maximo Cobos[1] · Francesc J. Ferri[1]

Accepted: 28 November 2024

Abstract

This paper addresses road safety concerns by investigating low-cost solutions for sound event detection (SED) tailored to driving scenarios. While advanced technologies like deep learning hold promise for improving road safety, their practical implementation often involves expensive sensors and hardware. Distractions, a major cause of accidents, require effective detection and mitigation. This study concentrates on auditory distractions and utilizes SED with low-cost edge devices to identify and timestamp relevant audio events, providing valuable insights into the driving environment. We evaluate state-of-the-art deep learning models on various edge devices, including the 2023 DCASE baseline with convolutional recurrent neural networks (CRNN) and an adapted YOLO vision model for audio spectrograms. Our analysis spans different hardware options, including single-board computers (SBCs) and desktop equipment, offering guidance on cost-effective hardware selection for in-vehicle SED applications. This research aims to contribute to affordable SED solutions in the context of driving safety, with the ultimate goal of advancing road safety efforts worldwide.

Keywords Sound event detection · Deep learning · Driver safety · Smart vehicles · Edge computing platforms

✉ Maximo Cobos
maximo.cobos@uv.es

Carlos Castorena
carlos.castorena@uv.es

Jesus Lopez-Ballester
jesus.lopez-ballester@uv.es

Juan A. De Rus
juan.rus@uv.es

Francesc J. Ferri
francesc.ferri@uv.es

[1] Universitat de València, Valencia, Spain

Journal : SmallExtended 11227	Article No : 6796	Pages : 15	MS Code : 6796	Dispatch : 5-12-2024

1 Introduction

Road safety stands as an ever-pressing concern, profoundly impacting lives and communities worldwide. The sustainable development goals (SDGs) have underscored the paramount importance of curbing traffic accidents, setting ambitious objectives to reduce fatalities and injuries significantly [1]. Despite positive trends, car accidents remain a prominent cause of global fatalities, demanding unwavering attention and innovative solutions. In response to these challenges, researchers have increasingly turned to technological advancements, including deep learning techniques, to enhance road safety. Autonomous vehicles, eye-tracking technology [2], and image processing with convolutional neural networks (CNNs) [3] have emerged as valuable tools for accident prevention. These technologies offer great promise, but their widespread adoption often hinges on the availability of costly and sophisticated sensors and hardware.

Distractions, a leading cause of accidents, can manifest in various forms, ranging from mobile phone usage to interactions with passengers [4–7]. Detecting and mitigating these distractions is vital for improving road safety. While advanced sensors and computing power are beneficial for this purpose, the cost of deploying such systems in every vehicle remains a significant hurdle. Our research delves into the detection of auditory distractions using sound event detection (SED) with low-cost devices. SED involves recognizing and timestamping audio events, providing insights into the driving environment and potential hazards. While SED has garnered attention in various domains, its application within driving safety, particularly with low-cost devices, is relatively underexplored.

This study aims to bridge this gap by focusing on low-cost SED solutions specifically tailored for driving scenarios. We analyze some state-of-the-art deep learning models implemented on edge devices, assessing their computational load and suitability for real-time processing, as well as their detection performance. More specifically, we consider the 2023 DCASE baseline based on a convolutional recurrent neural network (CRNN) and an adaptation of the well-known YOLO vision model working over audio spectrograms, both strategies were previously studied and compared for detecting distractors in driving environments in a prior work [8]. This work demonstrates the capabilities of these two models for the task and serves as the foundation for our current study. To provide a general view of the possibilities offered by edge devices, we analyze the performance of such models on a wide range of devices, including single-board computers (SBCs) and desktop equipment.

Additionally, the models have been evaluated after applying quantization techniques commonly used to improve inference times and model sizes without affecting their precision [3]. This is especially useful in devices with limited computational resources. The outcome of this work is expected to provide insights into the practicality of implementing state-of-the-art deep learning models on edge devices. We aim to offer guidance on selecting suitable hardware for affordable SED applications in vehicles, elucidating the range of options available for practical implementation and contributing to the advancement of cost-effective SED solutions within the context of driving safety.

Journal : SmallExtended 11227	Article No : 6796	Pages : 15	MS Code : 6796	Dispatch : 5-12-2024

The remainder of this paper is organized as follows: Sect. 2 introduces the auditory distractors relevant to driving scenarios and provides a detailed description of the dataset used for training and evaluation. Section 3 focuses on the sound event detection (SED) models employed, including their architecture and implementation details. Section 4 describes the experimental setup, covering the different hardware platforms and optimization techniques used in the study. The results of these experiments, including detection performance and computational metrics, are presented in Sect. 5. Finally, Sect. 6 offers conclusions and insights into the implications of our findings for real-world applications of SED in driving environments.

2 Auditory distractors and datasets

The auditory landscape experienced by drivers during vehicle operation encompasses a wide array of sound events. These acoustic occurrences emanate from diverse sources, including both internal elements within the vehicle, such as vehicle components, electronic devices, and human interactions, as well as external factors like traffic conditions, road surfaces, and environmental elements (refer to Fig. 1). In the context of SED for driving safety, the paramount objective is to discern and prioritize auditory distractions that possess the potential to substantially divert a driver's focus and thereby introduce road safety risks. The selection of these distraction categories is grounded in comprehensive literature review and prior research within the field [7, 9–14]. While an array of factors can contribute to diverting a driver's attention, this discussion specifically centers on well-recognized auditory distractions, including nine categories, namely, "Ring Tone," "Vibrating Phone," "Notification," "Speech," "Baby Cry," "Physiological" events (comprising sneezing, coughing, and yawning), "Pets" (encompassing sounds associated with cats and dogs), "Horns," and "Sirens." These distraction categories are all included in the taxonomy introduced in Castorena et al. for driving environments [8].

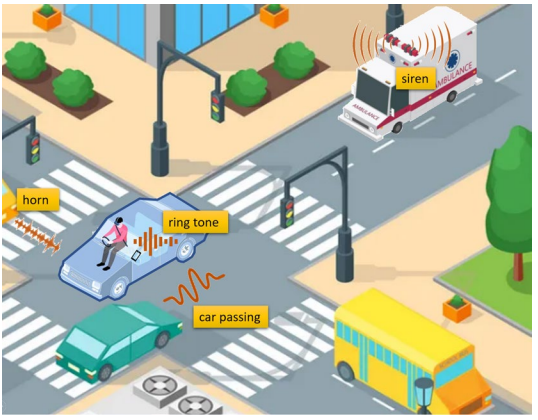


Fig. 1 Driver exposed both to internal (e.g., ring tone) and external sounds

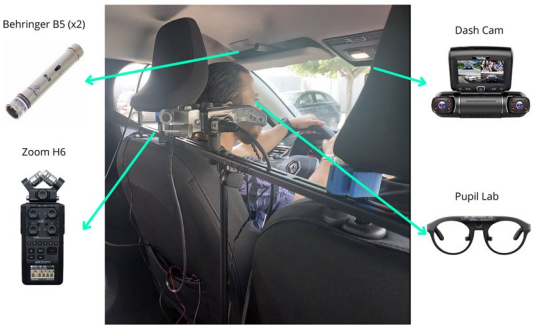


Fig. 2 Recording setup inside the cabin for the real dataset

2.1 Datasets

This study considers the recently released "SED for Driving Safety" dataset,[1] specifically designed to facilitate training and evaluation of SED systems in the context of driving safety. The focus of the dataset is on the identification of the above auditory distractors in synthetic audio recordings that simulate driving scenarios, including other common events that constitute the intricate acoustic landscape encountered during driving. The dataset holds 19,000 audio files, each lasting 10 s (totaling around 53 h). Of these, 15,000 are for training, while the rest are split evenly for validation and testing, with no overlap. Audio mixes were created to balance event occurrences.

In addition to the synthetic dataset, we collected 475 audio recordings, each 10 s long, under real-world driving conditions. Each segment contains a maximum of four events from different classes, with up to three overlapping classes. All nine classes are appropriately represented across the whole set of recordings. The recordings for this experiment were captured using a Behringer B5 microphone with a cardioid capsule, strategically placed to prioritize sound capture from the driver's perspective. Figure 2 shows a photograph of the configuration. Although multiple microphones distributed within the car cabin were employed, we only used the one closest to the driver, as our experiments focus on single-channel predictions. This setup offers high-quality audio capture, particularly effective at detecting sounds both inside and outside of the vehicle. The data collection took place inside a car with two different drivers, a copilot, and a passenger. To maintain the realism of the test, the drivers were unaware of the specific distractors being tested, while the passenger actively provoked the generation of various sound events. This approach allowed us to capture authentic interactions and responses to common auditory distractions. All the recorded audio samples were manually labeled, ensuring precise annotation of the events for subsequent model training and evaluation. The result, hereafter referred to as real dataset, was used exclusively for testing purposes.

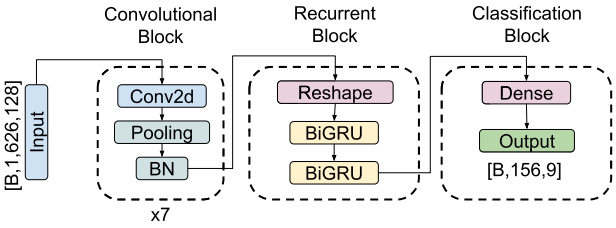
[1] https://www.kaggle.com/datasets/ccastorena/sound-event-detection-for-driver-safety.

Journal : SmallExtended 11227 | Article No : 6796 | Pages : 15 | MS Code : 6796 | Dispatch : 5-12-2024

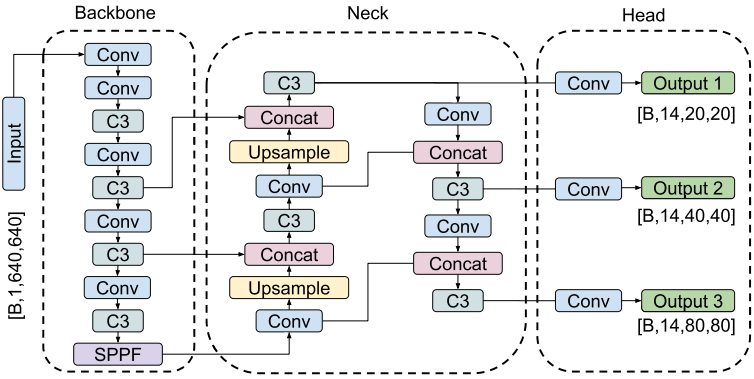
3 SED models

Deep learning, especially CRNN models, has proved effective for SED tasks. CRNNs extract features from spectrograms and capture temporal patterns using recurrent layers like LSTM or GRU. Our study replicates the DCASE 2021-2023 CRNN baseline [15], a model that includes seven convolutional blocks, each one with a 2D convolutional layer (Conv2d), batch normalization (BN), GLU activation, and max pooling layer (Pooling). Two bidirectional GRU layers (BiGRU) produce 156 frames with event activations, while the input is a Mel spectrogram of 128 bands and 626 time frames (10 s), as shown in Fig. 3a. Labels use a frame-by-frame format with hard (per frame) and weak (absence/presence in the whole output) labels. Training includes mixup augmentation, lasting 200 epochs, optimizing hard, weak, and consistency errors [16].

Additionally, this study uses a novel approach based on YOLOv5 presented in [8], where YOLO, originally designed for detection in images, is adapted to process audio. While previous work explored earlier versions of YOLO for SED, this approach extended it to audio analysis, drawing inspiration from Venkatesh et al. [17] approach but with variations. The model estimates sound event



(a) CRNN



(b) YOLO

Fig. 3 Summarized description of the models used

duration, midpoint, and employs distinct convolutional connections, enhancing accuracy in class activity predictions. YOLO's efficiency and scalability make it ideal for resource-constrained, real-time applications. Unlike CRNN, YOLO-based approach prioritizes convolutional networks, leveraging their parallel processing capabilities. The YOLOv5 architecture comprises three sections: backbone for feature extraction, neck for spatial pyramidal pooling, and head for bounding box predictions. Each section primarily consists of convolutional blocks (2D convolutional layers, batch normalization, SiLU activation function, and pooling layers), along with C3 blocks that incorporate a bottleneck module. This fully convolutional architecture is illustrated in Fig. 3b. We focus on smallest version: YOLOv5n (nano). The spectrogram input is padded to match the YOLO input shape (640×640). The head produces multi-scale outputs for events of varying sizes, with three types of losses for localization accuracy, classification correctness, and prediction confidence. Training consists of 200 epochs with a batch size of 48 samples. We use the mosaic technique for data augmentation, combining four spectrograms with random shifts. As a consequence, events are placed at arbitrary vertical positions within the 640×640 input, forcing the model to learn the meaning of frequency bands regardless of its absolute position.

Further details regarding the implementation of the SED models, including specific configurations, training procedures, architectural choices, techniques for data augmentation, post-processing, as well as the interpretation of model outputs, can be found in [8].

3.1 Model deployment

The CRNN and YOLO-based models used in this work are presented in the Open Neural Network Exchange (ONNX) format [18], which provides a standardized representation of models without the dependency on the framework used during training, maintaining their performance by avoiding optimization methods during transformation. This open ecosystem can run on numerous devices, including edge devices, as well as on multiple platforms and languages [19, 20]. In this study, we have selected the Python language for implementation and execution on four different processors. The first three processors are considered edge devices: the first one is an Arm Cortex-A57 found in a Jetson Nano, operating at a maximum frequency of 1.43 GHz and maximum input power of 10.0W according to the data sheet; the second device is a Raspberry Pi 4 with a Cortex-A72 processor, clocked at 1.80 GHz and 15.0W of maximum input power; the third processor is an Intel Pentium N3710, mounted on a UDOO X86 board, with four cores, a maximum frequency of 2.56 GHz and maximum input voltage of 12V and 3A (36W). These devices exemplify typical edge computing platforms. Lastly, the fourth processor is an Intel Core-i7 with eight cores and a frequency of 3.60 GHz, used as a reference platform but not considered an edge device. A summary of these devices can be found in Table 1.

Journal : SmallExtended 11227	Article No : 6796	Pages : 15	MS Code : 6796	Dispatch : 5-12-2024

Table 1 Summary of hardware devices and their specifications used for model implementation and execution

Name Device	Name CPU	Num Cores	Freq (GHZ)	RAM (GB)	Power (W)
Jetson Nano	Arm Cortex-A57	4	1.43	4	10
Raspberry Pi 4	Arm V8 Cortex-A72	4	1.80	4	15
UDOO X86 II Ultra	Intel Pentium N3710	4	2.56	8	36
PC	Intel Core-i7 9700k	8	3.60	16	900

3.2 Model compression techniques

In order to deploy deep learning models effectively on edge devices with limited computational resources, optimization techniques are essential to reduce the size of the models and improve their inference speed without significantly compromising accuracy. Three of the most popular optimization methods applied in the state of the art [19, 21–25] are model pruning, static quantization, and dynamic quantization.

Pruning typically reduces the number of weights by removing specific model parameters but typically offer worst results that quantization [21, 25], after preliminary tests on our data, pruning was not applied in this study due to its limited impact on performance, as it showed no significant improvements in either prediction times or model size for the architectures used in this work.

Quantization is a process that converts model weights and activations from 32-bit floating point (FP32) to a lower precision format, typically 8-bit integers (INT8). This reduction in precision leads to faster computations and lower memory usage while maintaining minimal accuracy loss. In this study, we applied both static quantization and dynamic quantization. Static quantization involves converting both the weights and activations of the model to INT8 format ahead of time, during a calibration phase that uses a representative dataset to determine the quantization parameters [24], such as scale and zero-point, to optimize inference performance. Dynamic quantization, in contrast, only converts the weights to INT8 format ahead of time, while the activations remain in FP32 during inference and are quantized dynamically at runtime [24]. This approach provides greater flexibility and slightly higher accuracy compared to static quantization, though with a minor trade-off in computational efficiency. The implementation of these techniques was carried out using ONNX [18], as it provides a flexible and efficient framework for deploying models across different hardware platforms.

4 Experiments

Conducting comprehensive experiments to evaluate both the prediction performance of our models and their computational profile on various platforms is of paramount importance from a practical deployment perspective. These experiments allow us

Journal : SmallExtended 11227 | Article No : 6796 | Pages : 15 | MS Code : 6796 | Dispatch : 5-12-2024

to gain insights into the real-world feasibility of low-cost SED solutions tailored for driving scenarios. Prediction performance assessment provides us with a clear understanding of how accurately our models can detect and classify sound events, directly impacting further actions to enhance road safety. Additionally, examining prediction times on different platforms is crucial for assessing the computational efficiency of the considered models, enabling us to determine their suitability for real-time processing.

RAM memory usage is another critical factor to consider, as it directly impacts the resource requirements of the models, ensuring they can operate effectively within the constraints of various hardware configurations. Lastly, monitoring processor temperature provides insights into the thermal management of these platforms during SED tasks, which is essential for reliability and safety during extended vehicle use. By studying these parameters comprehensively, we can make well-informed decisions about hardware selection and model optimization, ultimately contributing to the development of cost-effective and reliable SED solutions to improve road safety.

5 Results

5.1 Detection performance

The detection performance of both models was evaluated on the test partitions of synthetic and real datasets, leading to the results shown in Table 2, which considers two event-based metrics. On the one hand, we calculated the polyphonic sound event detection scores (PSDS) provided by the `sed_eval` toolbox [26, 27], a recognized reference in the DCASE community. PSDS measures the ability of a system to detect sound events in a given audio, considering as true/false positives the events based on the detection tolerance criterion (DTC), the ground-truth intersection criterion (GTC), and other important parameters. Specifically, we explored PSDS ($DTC = GTC = 0.5$) configuration. Similarly, we evaluated the segment-based class-averaged F_1, which assigns true/false positives frame by frame, considering as true positives the frames with the presence of an event and as true negatives those

Table 2 Detection performance of YOLO and CRNN models with and without static and dynamic quantization

		Synthetic		Real	
		F1	PSDS	F1	PSDS
YOLO	No Q	0.75	0.56	0.81	0.36
	Dynamic Q	0.74	0.50	0.80	0.35
	Static Q	0.74	0.48	0.80	0.32
CRNN	No Q	0.65	0.37	0.77	0.14
	Dynamic Q	0.58	0.10	0.76	0.15
	Static Q	0.63	0.13	0.76	0.04

Journal : SmallExtended 11227	Article No : 6796	Pages : 15	MS Code : 6796	Dispatch : 5-12-2024

with the absence of the event class. For this study, a frame resolution of 0.1 s was considered.

Moreover, we evaluated the performance of both models under quantization to assess its impact on detection in the test partition of the real dataset. The results, shown in Table 2, indicate a slight reduction in performance when quantization is applied, particularly in static quantization, where both models exhibit lower PSDS scores compared to the non-quantized versions. For instance, in real dataset, the YOLO model with static quantization achieves an F1 score of 0.80 and a PSDS of 0.32, slightly lower than its non-quantized counterpart. Similarly, the CRNN model shows a more pronounced decrease in PSDS dropping from 0.14 without quantization to 0.04 with static quantization. Note that the F1 score for synthetic and real datasets cannot be directly compared as it is heavily influenced by event density which is significantly lower for our real audios.

It is evident that quantization has a greater impact on the CRNN model compared to YOLO, particularly when considering the PSDS metric. This is primarily due to the large penalty imposed by this metric on false positives, which are amplified after quantization when existing events are split into multiple segments due to the higher granularity output of the CRNN model. To illustrate this, Fig. 4 shows the true labels and the predictions made by the different model configurations on two selected inputs that are representative of different behaviors. While in sample 200, the predictions remain largely unaffected after applying the compression techniques, in sample 37, CRNN predictions shift from displaying a single false-positive

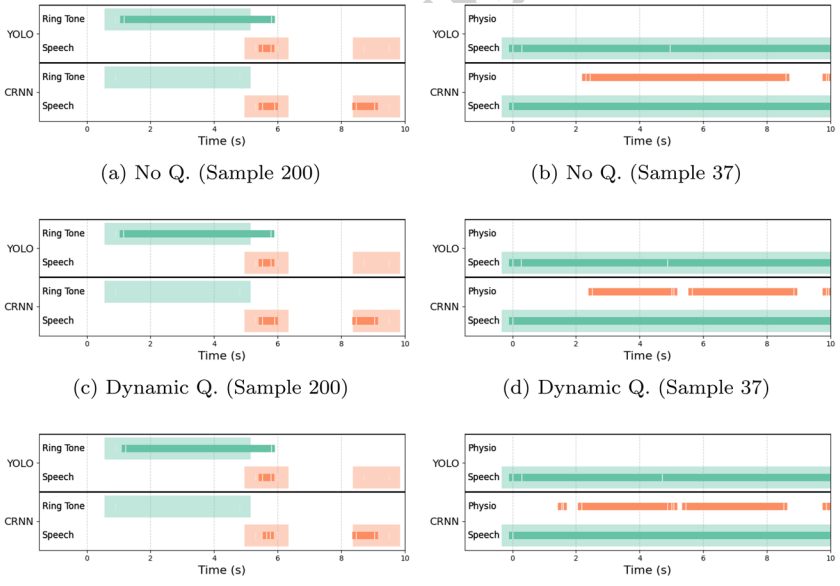


Fig. 4 Graphical representation of the ground truth (lighter, wider bars) and predictions (solid, narrower bars) by YOLO and CRNN models with and without static and dynamic quantization for two different input samples

Journal : SmallExtended 11227	Article No : 6796	Pages : 15	MS Code : 6796	Dispatch : 5-12-2024

event in the physiological class to multiple false positives after compression. This division, in turn, justifies the small loss in the segment-based F1 score, as under this metric only the frames that remain active even if the event is segmented are penalized.

5.2 Computational performance

To assess computational performance effectively, we deployed a system that processes 30 min of recorded audio, resulting in 180 measurements, one for each 10-second audio segment. In this setup, the inference is not performed continuously; instead, we simulate real-time conditions where the system records each 10-second segment while simultaneously performing the prediction on the previous segment in parallel. This approach aims to replicate real-world scenarios, where recording and prediction occur concurrently. The evaluation focuses on measuring key aspects such as Mel spectrogram computation, preprocessing, inference, and post-processing.

It is important to note that some of the measured aspects, such as Mel spectrogram computation, preprocessing, and post-processing of the output, are much less influenced by the model compression method used. These processes remain relatively consistent regardless of whether static or dynamic quantization is applied. However, metrics such as inference time and processor temperature are directly influenced by the compression technique employed. The type of quantization affects how efficiently the model operates during inference, leading to variations in speed and thermal load depending on the method used.

The inference times for both models, YOLO and CRNN, are presented in the accompanying Fig. 5, which displays a standard boxplot. The figure compares the models with and without the application of static and dynamic quantization, and

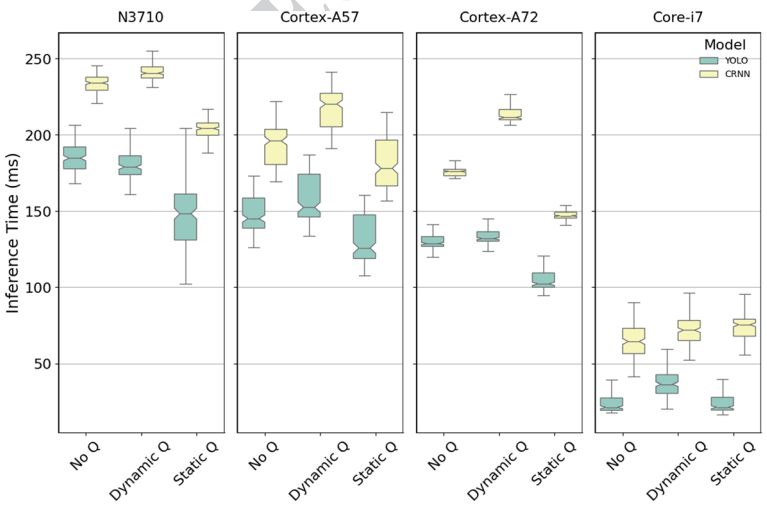


Fig. 5 Standard boxplots of inference time when deploying CRNN and YOLO models on different hardware platforms and compressing methods

Journal : SmallExtended 11227	Article No : 6796	Pages : 15	MS Code : 6796	Dispatch : 5-12-2024

each column in the plot represents a different hardware platform. Higher values in the plot indicate that the model is slower, while lower values reflect better performance in terms of speed, measured in milliseconds.

Figure 5 clearly demonstrates that across all platforms, the YOLO model significantly outperforms the CRNN model in terms of inference time. This is particularly noteworthy given that YOLO has a larger number of parameters (1.9 million compared to 1.1 million for CRNN), highlighting the efficiency of the model despite its higher complexity. One possible explanation for the superior efficiency of YOLO lies in its architecture, which relies exclusively on convolutional layers that are highly parallelizable. This design enables YOLO to fully leverage modern hardware acceleration, as convolutional operations are well optimized on both CPUs and GPUs, allowing for efficient processing on most processors. It is also worth noting that on platforms with more powerful processors, such as the i7, the difference in performance between the two models is reduced, suggesting that higher computational resources mitigate the gap between the two architectures.

When dynamic quantization is applied, inference times tend to increase, as expected, due to the overhead introduced by dynamically converting activations during inference. This effect is consistent across all platforms, though it has a greater impact on the CRNN model compared to YOLO. In contrast, the best results are achieved with static quantization, which not only improves inference times relative to the non-quantized models but also provides consistent speed gains for both YOLO and CRNN across all platforms. These results suggest that model compression alone is not sufficient to achieve low inference times. A careful selection of models is also crucial, as even the best case for the CRNN model with static quantization is slower than the non-quantized YOLO model. This underscores the importance of model architecture in optimizing real-time performance, particularly in resource-constrained environments.

Figure 6 presents a standard boxplot that visualizes the times for Mel spectrogram computation, preprocessing, and post-processing. Unlike the previous figure, the results for models with and without quantization have been aggregated, as these measured times behaved very similarly regardless of model compression. This allows us to present a unified comparison of the different subprocesses across the platforms. Notably, the Mel spectrogram computation consumes almost half of the total inference time, making it the second most computationally expensive subprocess. This highlights its importance in the overall performance of the system. In terms of preprocessing, the YOLO model is slightly slower due to the additional padding required to make the input compatible with its architecture. However, the difference between YOLO and CRNN in preprocessing time is minimal, never exceeding 1 millisecond in any case. For post-processing, which involves transforming the model outputs into final predictions (onset, offset, and class), YOLO also exhibits longer times and greater variability. This is largely due to the number of predicted events, as the model must iterate through each one to remove overlapping events. This additional complexity results in slightly higher post-processing times compared to CRNN.

Table 3 summarizes the temperature and RAM usage of the YOLO and CRNN models across different hardware platforms, both with and without quantization.

Journal : SmallExtended 11227 | Article No : 6796 | Pages : 15 | MS Code : 6796 | Dispatch : 5-12-2024

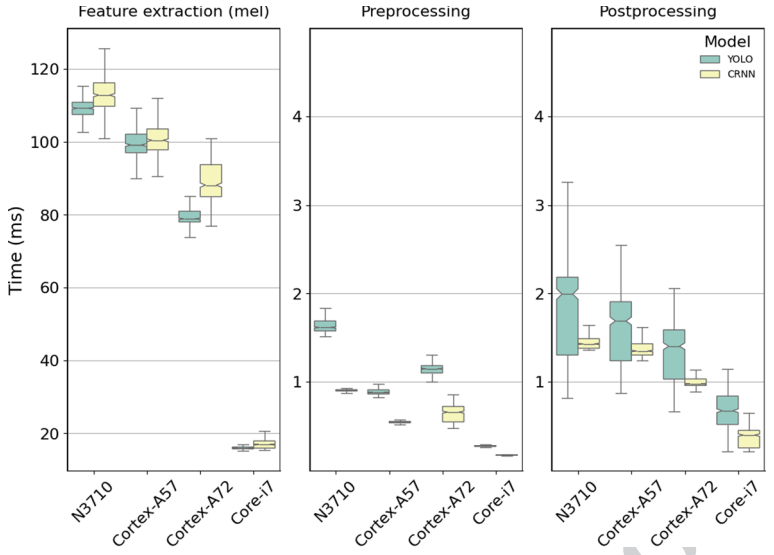


Fig. 6 Standard boxplots of feature extraction, pre- and post-processing times when deploying CRNN and YOLO models on different hardware platforms

Table 3 Mean values and standard deviations were appropriate, corresponding to memory (RAM) usage and operating temperature when using CRNN and YOLO models on the different hardware platforms considered

			No Q	Dynamic Q	Static Q
Temperature (°C)	YOLO	N3710	58.44 ± 1.35	62.00 ± 1.26	56.53 ± 1.49
		Cortex-A57	32.09 ± 0.54	32.04 ± 0.48	31.26 ± 0.50
		Cortex-A72	51.25 ± 0.51	50.94 ± 0.54	50.45 ± 0.55
		Core-i7	51.53 ± 0.67	44.00 ± 0.76	51.25 ± 1.35
	CRNN	N3710	52.18 ± 0.54	51.46 ± 1.02	52.56 ± 0.55
		Cortex-A57	32.29 ± 0.69	30.93 ± 0.74	30.59 ± 0.86
		Cortex-A72	51.25 ± 0.51	50.94 ± 0.54	50.45 ± 0.55
		Core-i7	53.23 ± 0.84	52.36 ± 1.21	52.51 ± 0.72
RAM (MB)	YOLO	N3710	225.56 ± 0.48	221.10 ± 0.72	223.06 ± 0.46
		Cortex-A57	229.15 ± 1.49	223.39 ± 0.97	227.94 ± 0.82
		Cortex-A72	244.14 ± 0.85	240.07 ± 0.72	239.39 ± 0.71
		Core-i7	250.08 ± 0.46	243.95 ± 0.70	248.25 ± 0.71
	CRNN	N3710	257.06 ± 2.34	260.61 ± 2.34	229.88 ± 0.78
		Cortex-A57	262.74 ± 2.55	264.40 ± 2.35	230.66 ± 0.71
		Cortex-A72	278.95 ± 2.36	276.87 ± 2.34	243.65 ± 1.48
		Core-i7	286.15 ± 2.33	286.64 ± 2.37	253.88 ± 0.38
Model size (MB)	YOLO	All	9.63	2.60	2.60
	CRNN	All	4.25	2.52	2.52

Journal : SmallExtended 11227 | Article No : 6796 | Pages : 15 | MS Code : 6796 | Dispatch : 5-12-2024

Notably, the Cortex-A57 platform maintains relatively low temperatures across all configurations, while the N3710, Cortex-A72, and Core-i7 platforms show the highest temperatures, particularly for YOLO with dynamic quantization on N3710 processor, which reaches an average of 62.0 °C. In terms of RAM usage, CRNN consistently uses more memory than YOLO across all platforms. The i7 processor shows the highest RAM usage for both models, peaking at 286 MB for the non-quantized CRNN and 250 MB in YOLO. However, with static quantization, RAM usage decreases significantly, especially for CRNN, where the memory consumption drops by more than 30 MB on most platforms.

Finally, model size reduction was significant with quantization. The YOLO model, originally sized at 9.63 MB, was reduced to 2.60 MB with both dynamic quantization and static quantization. Similarly, the CRNN model, which initially had a size of 4.25 MB, was reduced to 2.52 MB after applying quantization techniques.

6 Conclusion

This work has focused on evaluating the promising prospects of low-cost sound event detection (SED) solutions for driving scenarios. As road safety remains a critical global concern, the need for efficient detection and mitigation of distractions in a cost-effective manner becomes increasingly evident. Our investigation into auditory distractions using state-of-the-art deep learning models, implemented on various edge devices, has yielded valuable insights. Notably, our experiments demonstrate the practical feasibility of these models, as they exhibit robust prediction performance across multiple platforms. The evaluation of prediction times, RAM memory usage, and processor temperature further underscores their applicability for real-time processing in resource-constrained environments. Such comprehensive analyses serve as a practical guide for the deployment of SED solutions in vehicles, contributing to improved road safety. This research not only advances our understanding of cost-effective SED but also paves the way for its wider adoption.

Acknowledgements This work has been supported by Grant TED2021-131003B-C21 funded by MCIN/AEI/10.13039/501100011033 and by the "EU Union NextGenerationEU/PRTR", as well as by Grant PID2022-137048OB-C41 funded by MICIU/AEI/10.13039/501100011033 and "ERDF A way of making Europe". Authors would like also to thank *Generalitat Valenciana-Santiago Grisolía* program for financing this work (GRISOLIAP/2021/060, CPI-21-232). Finally, the authors acknowledge as well the Artemisa computer resources funded by the EU ERDF and Comunitat Valenciana, and the technical support of IFIC (CSIC-UV).

Author contributions Carlos Castorena conceived the idea, conducted the experiments, and wrote the main manuscript. Jesus Lopez-Ballester provided technical and experimental assistance. Juan Antonio de Rus contributed to the planning and execution of the experiments. Maximo Cobos and Francesc J. Ferri provided technical supervision and assisted in the writing and analysis of the results. All authors reviewed and approved the final manuscript.

Data availability No datasets were generated or analyzed during the current study.

Declarations

Conflict of interests The authors declare no competing interests.

Journal : SmallExtended 11227	Article No : 6796	Pages : 15	MS Code : 6796	Dispatch : 5-12-2024

References

1. The sustainable development goals report 2022, July 2022. [Online]. Available: https://unstats.un.org/sdgs/report/2022/
2. Vicente F, Huang Z, Xiong X, la Torre F, Zhang W, Levi D (2015) Driver gaze tracking and eyes off the road detection system. IEEE Trans Intell Transp Syst 16:2014–2027
3. Li W, Huang J, Xie G, Karray F, Li R (2021) A survey on vision-based driver distraction analysis. J Syst Architect 121:102319
4. Li W, Gkritza K, Albrecht C (2014) The culture of distracted driving: evidence from a public opinion survey in IOWA. Transp Res F: Traffic Psychol Behav 26:337–347
5. Prat F, Planes M, Gras ME, Sullman MJ (2015) An observational study of driving distractions on urban roads Spain. Accid Anal Prevent 74:8–16
6. Prat F, Gras ME, Planes M, Font-Mayolas S, Sullman MJ (2017) Driving distractions: an insight gained from roadside interviews on their prevalence and factors associated with driver distraction. Transp Res F: Traffic Psychol Behav 45:194–207
7. Farmer CM, Braitman KA, Lund AK (2010) Cell phone use while driving and attributable crash risk. Traffic Inj Prev 11(5):466–470
8. Castorena C, Cobos M, Lopez-Ballester J, Ferri FJ (2024) A safety-oriented framework for sound event detection in driving scenarios. Appl Acoust 215:109719
9. Edwards S, Wundersitz LN, Australia S (2019) Distracted driving: prevalence and motivations. Accid Anal Prevent 54:99–107
10. Koppel S, Charlton J, Kopinathan C, Taranto D (2011) Are child occupants a significant source of driving distraction? Accid Anal Prevent 43(3):1236–1244
11. Regan MA, Oviedo-Trespalacios O (2022) Driver distraction: mechanisms, evidence, prevention, and mitigation. In: The Vision Zero Handbook: Theory, Technology and Management for a Zero Casualty Policy. (pp 995–1056). Springer International Publishing, Cham
12. Zou T, Guo H, Khaloei M, MacKenzie D, Boyle LN (2023) Examining the relationships between multimodal environments and multitasking driving behaviors. Transp Res Rec 2677(2):944–957
13. Nagahama A, Tanaka K, Feliciani C, Cui G, Wada T (2022) Effects of urban landscape and soundscape on driving behavior. In: 2022 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA). (pp 84–88). IEEE
14. Prohn MJ, Herbig B (2023) Potentially critical driving situations during “Blue-light" driving: a video analysis. West J Emerg Med 24(2):348
15. Turpault N, Serizel R, Shah AP, Salamon J (2019) Sound event detection in domestic environments with weakly labeled data and soundscape synthesis. In: Workshop on Detection and Classification of Acoustic Scenes and Events
16. Tarvainen A, Valpola H (2017) Mean teachers are better role models: weight-averaged consistency targets improve semi-supervised deep learning results. Advances in neural information processing systems. 30
17. Venkatesh S, Moffat D, Miranda ER (2022) You only hear once: a YOLO-like algorithm for audio segmentation and sound event detection. Appl Sci 12:3293
18. Bai J, Lu, F and Zhang K. “ONNX: Open neural network exchange GitHub.” [Online]. Available: https://github.com/onnx/onnx
19. Ahn H, Chen T, Alnaasan N, Shafi A, Abduljabbar M, Subramoni H, Panda DK (2023) Performance characterization of using quantization for dnn inference on edge devices. In2023 IEEE 7th International Conference on Fog and Edge Computing (ICFEC). (pp 1–6). IEEE
20. Jin T, Bercea GT, Le TD, Chen T, Su G, Imai H, Negishi Y, Leu A, O’Brien K, Kawachiya K, Eichenberger AE (2020) Compiling ONNX neural network models using mlir. arXiv preprint arXiv:2008.08272.
21. Han S, Mao H, Dally WJ (2015) Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149.
22. Cerutti G, Prasad R, Brutti A, Farella E (2020) Compact recurrent neural networks for acoustic event detection on low-energy low-complexity platforms. IEEE J Select Topics Signal Process 14(4):654–664
23. Liang T, Glossner J, Wang L, Shi S, Zhang X (2021) Pruning and quantization for deep neural network acceleration: A survey. Neurocomputing 461:370–403

24. Rokh B, Azarpeyvand A, Khanteymoori A (2023) A comprehensive survey on model quantization for deep neural networks in image classification. ACM Trans Intell Syst Technol 14(6):1–50
25. Kuzmin A, Nagel M, Van Baalen M, Behboodi A, Blankevoort T (2023) Pruning vs quantization: Which is better? Adv Neural Inform Process Syst 36:62414–62427
26. Bilen Ç, Ferroni G, Tuveri F, Azcarreta J, Krstulović S (2020) A framework for the robust evaluation of sound event detection. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). (pp 61–65). IEEE
27. Ebbers J, Haeb-Umbach R, Serizel R (2022) Threshold independent evaluation of sound event detection scores. In: ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). (pp 1021–1025). IEEE