Artificial Intelligence Review (2025) 58:163
https://doi.org/10.1007/s10462-025-11106-z

Environmental sound recognition on embedded devices using deep learning: a review

Pau Gairí[1] · Tomàs Pallejà[1] · Marcel Tresanchez[1]

Accepted: 7 January 2025

Abstract
Sound recognition has a wide range of applications beyond speech and music, including environmental monitoring, sound source classification, mechanical fault diagnosis, audio fingerprinting, and event detection. These applications often require real-time data processing, making them well-suited for embedded systems. However, embedded devices face significant challenges due to limited computational power, memory, and low power consumption. Despite these constraints, achieving high performance in environmental sound recognition typically requires complex algorithms. Deep Learning models have demonstrated high accuracy on existing datasets, making them a popular choice for such tasks. However, these models are resource-intensive, posing challenges for real-time edge applications. This paper presents a comprehensive review of integrating Deep Learning models into embedded systems, examining their state-of-the-art applications, key components, and steps involved. It also explores strategies to optimise performance in resource-constrained environments through a comparison of various implementation approaches such as knowledge distillation, pruning, and quantization, with studies achieving a reduction in complexity of up to 97% compared to the unoptimized model. Overall, we conclude that in spite of the availability of lightweight deep learning models, input features, and compression techniques, their integration into low-resource devices, such as microcontrollers, remains limited. Furthermore, more complex tasks, such as general sound classification, especially with expanded frequency bands and real-time operation have yet to be effectively implemented on these devices. These findings highlight the need for a standardised research framework to evaluate these technologies applied to resource-constrained devices, and for further development to realise the wide range of potential applications.

Keywords Sound recognition · Audio classification · Deep learning techniques · Edge device · Real-time sensing · Resource-constrained devices

Abbreviations

ADC	Analog to digital converter
AI	Artificial intelligence

Extended author information available on the last page of the article

Published online: 15 March 2025

AUC	Area under the curve
CNN	Convolutional neural network
CPU	Central processing unit
CRNN	Convolutional recursive neural network
DCT	Discrete cosine transform
DL	Deep learning
DMA	Direct memory address
DSP	Digital signal processing
ESR	Environmental sound recognition
FCN	Fully convolutional network
FFT	Fast fourier transform
FLOPs	Floating point operations
GMM	Gaussian-mixture-model
GPU	Graphical processing unit
I2C	Inter-integrated circuit
I2S	Inter-integrated circuit sound
IDFT	Discrete fourier transform
LSTM	Long short-term memory
MACCs	Multiply-accumulate operations
MCU	Microcontroller
MEMS	Micro-electro-mechanical system
MFCC	Mel-frequency cepstral coefficient
MIPS	Million instructions per second
MOPS	Millions of operations per second
NVM	Non-volatile memory
PC	Personal computer
PCM	Pulse-code modulation
PDM	Pulse density modulation
RAM	Random-access memory
ResNet	Residual neural network
RNN	Recurrent neural network
SFTF	Short time fourier transform
SIMD	Single instruction multiple data
USB	Universal serial bus
KD	Knowledge distillation

1 Introduction

Acoustic vibrations serve as a pervasive medium for transmitting information in environments where human activity occurs. Unlike visual signals, sound offers several advantages in sensory perception, including the ability to capture blind spots through complete spatial acquisition, relatively low-cost equipment requirements, and minimal computational demands (Xiong et al. 2022). These factors make sound recognition a crucial component in achieving comprehensive digital sensing of the surrounding environment. Traditionally, sound information has been analysed from three primary domains: speech recognition, music recognition, and ESR (Sharma et al. 2020).

Environmental sounds exhibit structural differences compared to speech and music. They often lack the regular patterns or substructures characteristic of speech and music, instead presenting a more random, unpredictable nature (Chachada and Kuo 2014). Additionally, the inherent challenges of ESR are compounded by factors such as low signal-to-noise ratio, varying distances from sound sources, and the frequent overlap of multiple sound sources within a given environment (Bansal and Garg 2022). These characteristics underscore the distinct nature of ESR as a unique subfield within the broader discipline of digital information processing, requiring specialized methodologies and techniques.

A comprehensive review of the challenges in ESR is presented in (Chandrakala and Jayalakshmi 2019), which highlights issues such as the detection of multiple events from a single environment, the absence of a complete event dictionary in many cases, difficulties in recognition within complex scenes, the presence of background noise, the recurrence of identical audio events across different environments, and the lack of standardized, multimodal datasets.

Although ESR is not a widely explored field (Turchet et al. 2020), it holds a broad spectrum of both current and potential applications. Recent studies have begun to identify the unique characteristics of various sound sources and environments, thereby emphasizing the depth and complexity of this research area and contributing to the development of its taxonomy. For instance, audio event recognition has been employed in applications such as audio surveillance, audio fingerprinting, and audio spoofing, as discussed in (Prashanth et al. 2024), Further, studies focusing on the classification and processing of urban environmental sounds are summarized in (Nogueira et al. 2022a), while techniques for detecting natural events and recognizing animal species in natural settings are reviewed in (Meedeniya et al. 2023). Additionally, mechanical fault diagnosis and safety enhancement in industrial settings have been explored in (Tang et al. 2023), highlighting the diverse practical applications of ESR.

Given the inherent challenges of ESR and the additional requirements associated with every specific application, further research is essential to enable the successful integration of these potential applications into devices. Traditionally, these problems have been addressed by extracting specific features from the sound data, generating a reduced representation space of the information, and applying statistical or machine learning methods to segment this space in the desired manner (Babaee et al. 2017). Research has evolved toward methods that can extract relevant information directly from large amounts of data, known as DL (Purwins et al. 2019). Several studies address the comparison between these methods concluding better performance and accuracy for the DL approaches (Chandrakala and Jayalakshmi 2019) (Bansal and Garg 2022) (Mohammad and Tripathi 2019), even defining DL as the state-of-the-art in ESR (Mohaimenuzzaman et al. 2023) specially in terms of generalization and segmentation of large number of classes. Note that as the complexity of the task increases and the representation space expands, traditional models suffer from sparsity problems, making the ability of deep learning algorithms to learn more abstract representations particularly valuable (Li et al. 2017). Notably, in this context, CNNs and RNNs have been widely adopted as effective solutions for this purpose (Prashanth et al. 2024). Recent advancements have also focused on implementing more sophisticated DL techniques, such as Transformers (Nogueira et al. 2022b), MLP-Mixers (Tripathi and Pandey 2023), and Autoencoders (Libal and Biernacki 2024), to further enhance model accuracy. These approaches represent the forefront of DL research, having demonstrated notable success in other domains, including Computer Vision (Tolstikhin et al. 2021), Natural Language Processing (Vaswani et al. 2023), and Generative AI (Zhang et al. 2020). However, it is important to note that these models not only demand substantial computational resources for training but also require significant memory and floating-point operations to perform inference tasks (Canziani et al. 2017).

Most studies in the field of ESR focus primarily on signal processing and classification. However, many applications require or would significantly benefit from integrating the entire solution into embedded devices. The key reasons for this integration include:

(a) Seamless integration of the transducer, analog circuitry, and computational components of the system, such as with digital MEMS microphones (Zawawi et al. 2020).
(b) Reduction in device size, which is interesting for many practical applications (Hou et al. 2023), (Lin et al. 2024).
(c) Limited access to high-resource computing or power supply, as in devices intended for use in remote environments such as forests, farms, or marine settings (Meedeniya et al. 2023), (Huang et al. 2024).
(d) Need for wearable or portable devices, which demand efficient integration to ensure mobility and usability (He et al. 2022), (Hyun Choi et al. 2024).
(e) Multiple detection points to adequately characterize complex or large environments, benefiting from the deployment of Wireless Acoustic Sensor Networks (WASNs) and reducing data throughput by performing edge recognition computing (Hou et al. 2023), (Alsina-Pagès et al. 2020).

Processing sound requires significant sample rates, for example, the frequency range that humans can easily classify spans from 50 Hz to 15 kHz (Sharma et al. 2020). The bandwidth, along with the memory and computational demands of state-of-the-art ESR applications, combined with the requirement for real-time detection and edge device restrictions, pose significant challenges (Küçüktopcu et al. 2019) (Meedeniya et al. 2023). In this context, computing the DL algorithms in the cloud with data from the sensors is a possible approach. However, as noted in (Bahai 2024), performing the DL algorithms at the edge and transmitting only the results to the cloud can reduce communication bandwidth requirements, thereby enhancing overall network efficiency, data security, latency, reliability, and sensor power consumption.

This paper aims to provide an overview of current and emerging techniques and challenges in intelligent sound recognition applied to embedded devices, at the intersection of three scientific domains, as illustrated in Fig. 1: Intelligent Sound Recognition, DL Algorithms, and Embedded Devices. The main contributions of this paper are as follows:

(1) Emphasize and summarize the recent advances and integrations of ESR with DL in embedded systems as presented in the literature.
(2) Deconstruct the various modules and steps necessary for integrating sound recognition technology into embedded systems.

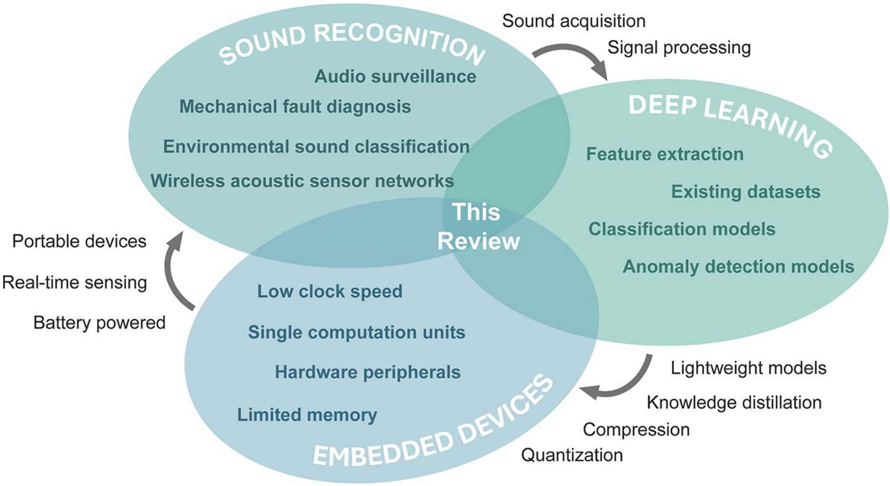

Fig. 1 This work links three science branches: sound recognition, deep learning algorithms, and embedded systems computation

(3) Analyse the existing possibilities for each module as presented in the literature, to provide a comprehensive guide to the various options available to designers in developing such solutions.
(4) Discuss the current state of the field and highlight directions for future work.

This review is organized as follows: Sect. 2 provides background on the potential applications of sound recognition, along with an analysis of the current state of edge implementation. Section 3.1 outlines the methodology of this work, while Sect. 3.2 discusses typical approaches for embedding DL algorithms. Section 3.3 evaluates input hardware and preprocessing algorithms, highlighting the key features commonly used for recognition tasks. A comparative analysis of DL algorithms is presented in Sects. 3.4, and 3.5 examines edge integration techniques, including methods for compressing and reducing model complexity, as well as specific lightweight models developed for edge applications.

2 Application background

In recent years, sound recognition through DL has emerged as a promising tool for various technological applications. Regarding industrial applications; in the construction industry, it has proven effective in detecting activities such as welding, cutting oriented strand boards, grinding, etc., (Xiong et al. 2022). In the woodworking industry, it can detect both the material being cut with a circular saw and the machine's power consumption (Svrzić et al. 2024). Additionally, sound recognition techniques are showing significant promise in detecting mechanical failures or early-stage issues in industrial equipment and machinery. As summarized in (Tang et al. 2023), several methods have been tested for diagnosing faults in rotating machinery, train bearings, and combustion engines. Beyond diagnostics, sound detection is increasingly seen as a safety tool for predicting industrial accidents. For example, in (Yu and Li 2023), a method is proposed for detecting gas or dust explosions in coal mines. Similarly, the safety and reliability of power systems are also being studied through sound recognition; in (Zhang et al. 2024) an acoustic-based method demonstrates advantages over existing vibration-based techniques for detecting faults in gas-insulated switchgear.

Furthermore, (Qurthobi et al. 2022) review the use of sound recognition in detecting industrial machine failures and highlight the potential of integrating failure detection into portable devices as a future direction to enhance the real-world applicability of this technology.

Beyond industrial applications, sound recognition has a wide range of applications. In wildlife detection and cataloguing, this technology plays an important role. The literature includes studies on the detection and classification of various species, ranging from bird species (Duan et al. 2024) (Yang et al. 2022a) (Han and Peng 2024), to insect species (Branding et al. 2024). The study in (Meedeniya et al. 2023) summarizes the methodologies of DL applied to the sound classification of the forest environment and exposes the benefits and challenges of integrating state-of-the-art approaches on edge devices. When real-time notifications are required, cloud-based solutions—where unprocessed data is sent to the cloud for computation— could be challenging due to the low infrastructure in these environments and the large areas involved. In such cases, an edge-based approach could provide a more efficient solution, allowing devices to report only the computed results, thereby offloading the network and enabling near real-time processing.

Another growing area of application for sound classification and detection using DL is the study of urban environments and the classification of associated events and activities. In (Nogueira et al. 2022a), systematic review exposes the different DL models and performance metrics highlighting the increasing academic interest in this field over the past decade. From a more specific application perspective, (Ciaburro and Iannace 2020) introduce a detection system for unmanned aerial vehicles (drones) that alerts for potential intrusion risks. The monitoring of human and social activities—both in urban environments and within homes—is reviewed in (Chandrakala and Jayalakshmi 2019) where several methodologies are compared to provide solutions for audio forensic analysis, road surveillance, fall detection or ambient assisted living.

Many of the above applications benefit from, or are only possible with, an embedded solution. However, research on the integration of these methods into embedded systems remains a small but growing field. Table 1 indexes and analyses recent publications related to embedding sound recognition applications using DL approaches. In this context, the precise level of integration of the solution is important, as the challenges, limitations, and potential of the techniques are directly related. In the literature, the term "on the edge" is used not only for embedded, resource-constrained devices such as MCUs but also for devices with greater computing resources, such as "MicroPCs" or complex SoCs like Raspberry Pi or smartphones. For this reason, a fundamental distinction is made in this work regarding the level of integration of the presented solutions:

1. **High level** of integration for solutions embedded and tested in a resource-constrained device or board using an MCU as a computational tool.
2. **Medium level** of integration for solutions tested on "MicroPcs" or similar portable devices, but with high computational resources.
3. **Low level** of integration for techniques or partial solutions intended and designed to be embedded on edge devices in future work. This last category is particularly relevant

Table 1 Analysed studies of specific implementation of sound recognition applications on the edge

References	Year	Application	Integration level	Device
(Montino and Pau 2019)	2019	Detection and classification of sounds emitted by car engines for urban traffic management. The application detects when a car is arriving or leaving the sensor area to provide a counting vehicle sensor. Provides a 3 class classification.	High	STM-32L476JG (MCU)
(Kumari et al. 2019)	2019	Classification of different events to achieve urban sound monitoring. Provides classification on existing datasets for 10 and 50 sound categories.	Low	“mote” device
(Cerutti et al. 2020)	2020	Classification of urban sound events using existing dataset. Provides classification for 10 classes.	High	STM32L-476RG (MCU)
(Naccari et al. 2020)	2020	Acoustic scene classification. Provides classification in 3 classes: indoor scene, outdoor scene or in-vehicle scene.	High	STM-32L476JG (MCU)
(Wyatt et al. 2021)	2021	Office sound classification in noisy environments. Provides classification for 6 classes.	Medium	Raspberry Pi Zero (MicroPC)
(Ko et al. 2022)	2022	Sound source detection and localization of human speech. Provides detection and localization in the horizontal plane of the speech source.	Medium	Raspberry Pi 4B (MicroPC)
(Choudhary et al. 2022)	2022	General environmental sound classification of 200 classes.	Medium	Samsung S21 Smartphone
(Strantzalis et al. 2022)	2022	Identification of operational states of a DC motor and diagnose of faulty conditions in real-time. Provides a 3 class classification of the motor sound.	High	STM32L4 Series (MCU)
(He et al. 2022)	2022	Human snore recognition in a sleep apnea preventing device. Provides 2 class classification.	High	STM32F767 (MCU)
(Mohaimenuzzaman et al. 2022)	2022	General environmental sound classification. Provides a 10 class classification.	Low	-
(Tripathi and Pandey 2023)	2023	Environmental sound classification. Provides a 10 class classification.	Low	-
(Mohaimenuzzaman et al. 2023)	2023	General environmental sound classification. Provides a 10 to 50 class classification.	Medium	Sony Spresense board - Sony CXD5602 (MCU)
(Laksono and Prasetio 2023)	2023	Speaker recognition used as a biometric authentication. Provides a 30 to 125 class classification.	Low	-
(Zhang et al. 2023)	2023	Domestic sound classification. Provides a 7 class classification.	Low	-
(Süer et al. 2023)	2023	Detection of failures in automotive manufacturing industries with the detection of sound emitted when plugging connectors. Provides an anomaly detection using reconstruction error of generative models.	Low	TIM Akilli Kiyafetleri A.S. Smart Gloves
(Somwong et al. 2023)	2023	Detection and classification of illegal activities in forest environment. Provides 4 class classification.	High	Arduino Portenta H7 (MCU)

Table 1 (continued)

References	Year	Application	Integration level	Device
(Marciniak et al. 2023)	2023	Monitoring of vacuum cleaner operational states. Classification of active state and different power levels. Provides 6 class classification.	High	Nordic Thingy:53, STM SensorTile.box, Arduino Nano 33 BLE Sense Lite (MCUs)
(Hammad et al. 2023)	2023	Anomaly detection in sound power levels of urban environments. Provides an anomaly detection using reconstruction error of generative models.	High	ESP32 S3 (MCU)
(Brighente et al. 2023)	2023	Drone detection for surveillance sentinel device. Provides 2 class classification.	High	Arduino Nano 33 BLE Sense (MCU)
(Maayah et al. 2023)	2023	Limit access to a car functions to limit children access. Provides detection of the key word and classification of the age or nature of the input voice with 4 class classification.	High	Arduino Nano 33 BLE Sense (MCU)
(Hou et al. 2023)	2023	Critical home event surveillance via detection of alarms or water falling unattended with a wireless device mesh. Provides 3 class classification.	High	STM32WB55 (MCU)
(Doinea et al. 2024)	2024	Health monitoring of bee hives. Detection of presence of queen bee with sound emissions near the beehive. Provides 2 class classification.	High	Arduino Nano 33 BLE Sense (MCU)
(Shi et al. 2024)	2024	Monitoring activity system for steel industries. Provides 7 class classification.	Low	-
(Priebe et al. 2024)	2024	Detection of human activity in natural environments trough the detection of speech. Provides binary classification.	Low	Portable device such Raspberry Pi
(Mou and Milanova 2024)	2024	General environmental sound classification. Provides a 10 to 50 class classification.	Medium	Raspberry Pi 4, NVIDIA Jetson Nano
(Hyun Choi et al. 2024)	2024	Sound recognition-based CPR training system. Classification of compression and depression sounds from the CPR training device. Provides a 2 class classification.	Medium	Samsung Galaxy Note 20 Smartphone
(Duan et al. 2024)	2024	Bird species detection and classification for wildlife monitoring. Provides a 20 class classification.	Low	-
(Wißbrock et al. 2024)	2024	Machinery fault detection in industry applied on fans, gears, pumps sliders and valves. Provides a binary detection of healthy or faulty equipment.	Low	-
(Wang et al. 2024)	2024	Detection of obstruction in vascular access using blood flow sound signals. Provides a 2 class classifier.	Low	-
(Libal and Biernacki 2024)	2024	Health monitoring of bee hives. Detection of presence of abnormal quantity of drone bees. Provides a binary classification.	Low	Custom LTE module.
(Sammarco et al. 2024)	2024	Detection of dangerous events in road environment such as driving cars, crashes, horns, tire explosions. Provides a 7 class classification.	Low	Smartphone
(Munirathinam and Vitek 2024)	2024	Emergency vehicle's siren detection and sound source localization for road traffic safety improvement. Provides a binary classification.	High	STM-32F411RE (MCU)

Table 1 (continued)

References	Year	Application	Integration level	Device
(Huang et al. 2024)	2024	Bird song recognition and bird specimen monitoring (Corn Bunting) in rural UK areas for sound datalogging improvements in wildlife monitoring. Provided with two class classifier	High	nRF52840 Development kit (MCU)
(Lin et al. 2024)	2024	Detection of hollow defect in tiles for building industry quality monitoring. Provided with two class classifier	High	Arduino Nano 33 BLE Sense (MCU)

because it points in the direction of embedding innovative solutions in very peripheral devices, although more research is needed to integrate the final solution.

As demonstrated by the analysis, there are a multitude of applications across a range of fields that employ the DL paradigm to derive valuable insights from acoustic media. Although some applications may appear similar or straightforward, the complexity of the solution is typically influenced by the nature of the sound data and the degree of generalization involved in the task. About classification complexity, for instance, the presented applications demonstrate a bias, defined by labels or distinct classes to identify, towards a higher level of integration, as evidenced by the fact that those with 50 or more distinction classes have a lower level of integration.

It is conceivable that with the implementation of optimisation strategies and the utilisation of dedicated tools from the embedded systems domain, the diverse background application sectors could reach the very edge paradigm and become technologies at the service of society. So is the aim of this work also to identify the successful techniques and the trending innovations to reach these objectives.

3 Methodology

3.1 Methodology of the study

This work has been organised around specific applications of sound recognition systems that are embedded, or designed to be embedded, in resource-constrained devices. A search of existing studies from the past five years has yielded Table 1. The following characteristics have been analysed in the selected studies:

1. Signal acquisition methods: hardware elements and specific acquisition techniques.
2. Preprocessing methods: signal sample rate, length of data, algorithms and parameters for the frequency domain translation.
3. Input features for the DL algorithm and their complexity.
4. DL approaches for different input domains or output tasks.
5. Optimization and compression techniques with accuracy, size and complexity.

All results have been presented in different straightforward tables to provide the lector with a broad overview of the field and its state of the art. Among the various characteristics discussed, a clear distinction is made regarding the level of integration and the different variables that may influence it. The paper presents a comprehensive guide, with proper references for each step in the application design, highlighting the different approaches linked to their respective integration levels.

3.2 Edge IA framework

As identified in (Saha et al. 2022) the Edge AI has a typical working framework that can be divided into two main tasks: the model development phase and the model deployment phase. In Fig. 2 the entire workflow is presented.

The model development phase is also referred to as the training process. It comprises several steps (Hou et al. 2023), (Maayah et al. 2023); beginning with the generation of a sufficient data collection or using of an existing dataset. Subsequently, a preprocessing strategy is determined, and the features used as input to the AI model are selected. Following this, a suitable DL algorithm approach is chosen, and the topology or the use of a well-established preexisting model is determined. The principal step of this phase is the training or fine-tuning of the DL algorithm to learn the characteristics of the task involved. The training of the model typically needs the processing of the dataset and segmentation to evaluate the performance metrics of the solution. These steps are conducted during the design of the solution, "offline" and with the associated computations being performed on a high-resource device, such as a PC or GPU.

The model deployment phase, as implemented, for example, in (Strantzalis et al. 2022) or (Cerutti et al. 2020), involves the application of optimization methodologies aimed at reducing computational complexity, cost, and model size. Computer-trained models typically utilise floating point variables; thus, a quantisation technique may be employed to obtain integer variables, thereby further optimising the size and computational cost. Ultimately, the

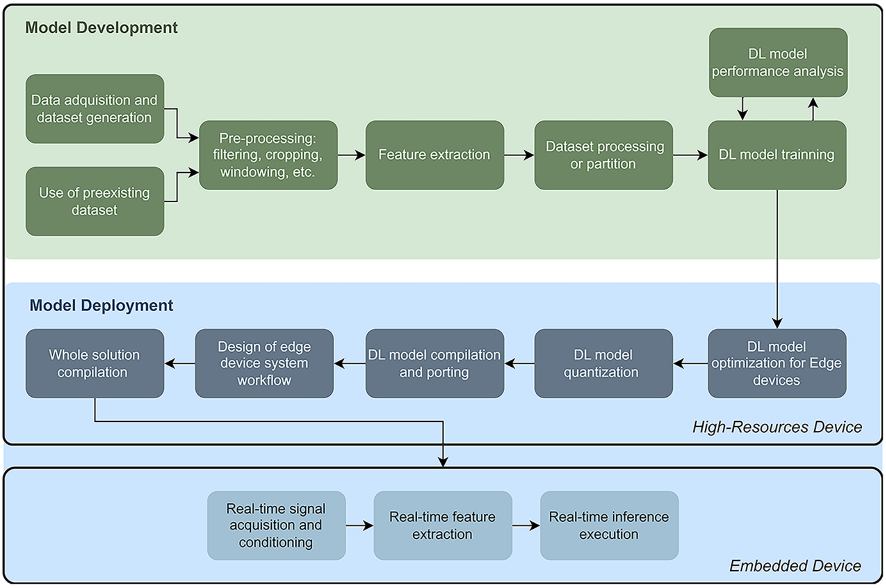
Model Development
DL model performance analysis
Data adquisition and dataset generation
Pre-processing: filtering, cropping, windowing, etc.
Feature extraction
Dataset processing or partition
DL model trainning
Use of preexisting dataset
Model Deployment
Whole solution compilation
Design of edge device system workflow
DL model compilation and porting
DL model quantization
DL model optimization for Edge devices
High-Resources Device
Real-time signal acquisition and conditioning
Real-time feature extraction
Real-time inference execution
Embedded Device

Fig. 2 Edge AI paradigm workflow

model can be transformed into an embedded system format, and a system workflow strategy can be developed and implemented to enable real-time data acquisition, processing, and inference calculation, while also optimizing energy consumption through techniques such as run-standby cadences or hardware accelerations. This final stage is integrated into the edge hardware.

3.3 From sound to data

One of the defining characteristics of edge devices is the integration of an entire system onto a single board. This characteristic necessitates considering the initial steps in the context of the sound recognition problem, from the acquisition of the signal from the physical world to the subsequent generation of coherent data on the input stream of the recognition algorithm. The following sections will trace the flow of information through the initial stages of the process.

3.3.1 Signal acquisition

The conversion of air vibrations into electrical signals is not typically addressed in studies of sound recognition techniques using DL. As a result, research tends to focus on pre-obtained signals or datasets, or even the signals obtained from unspecified systems, rather than the acquiring and conversion process itself. Table 2 presents an analysis of applications that utilise sound acquisition hardware or microphones. The data presented in the table demonstrate that MEMS technology is widely used. This can be attributed to the small footprint of the technology and the fact that the analog processing of the signal is conducted within the chip itself. For example, (Hou et al. 2023) used an ICS-43,434 MEMS omnidirectional microphone that provides direct digital audio data that is accessed through the I2S interface. Consequently, the system design is free from analog components, thereby enhancing its noise robustness. Conversely, despite the cost-effectiveness of MEMS microphones, their frequency response is typically limited from 100 Hz to 10 kHz, and they are therefore unsuitable for applications requiring a wider frequency range (Turchet et al. 2020).

Other approaches, which do not involve deep learning or edge integration, could also be considered. For example, in (Polo-Rodriguez et al. 2021) a low-cost microphone with a small form factor, not integrated with the edge device, is connected via USB. In (Küçüktopcu et al. 2019) an electret condenser microphone (CMA-4544PF-W) with an LM386 audio amplifier is used to obtain an analog signal, which is then converted through the ADC of the TI TivaC TM4C1294NCPDT MCU.

The combination of multiple microphones can improve the signal quality via hardware. For instance, in (Kim et al. 2021) an integrated array of four MP34DT01-M digital MEMS microphones is used, with a chip that incorporates DSP algorithms such as acoustic echo cancellation, beamforming, dereverberation, noise suppression, and gain control. This kind of microphone array could be used also to detect the sound source position as presented by (Ko et al. 2022) with a circular array of six MEMS microphones.

From an application perspective, microphones are not the only means of acquiring audio data. In (Wang et al. 2024), a stethoscope equipped with a condenser microphone is used to convert the vibrations within the skin into sound signals. Similarly, in (Shi et al. 2024), an industrial pickup is installed within the chassis of the equipment to capture the sound

Table 2 List of signal acquisition hardware for the embedded integrations analysed. All the microphones analysed presented an omnidirectional pattern

References	Microphone	Technology	Manufacturer	Footprint (mm)	SNR (dB)	Sensitivity (dBFS)	Signal
(Montino and Pau 2019; Naccari et al. 2020), (Brighente et al. 2023), (Maayah et al. 2023), (Doinea et al. 2024), (Lin et al. 2024)	MP34DT05-A	MEMS	STM	3 × 4 × 1	64	−26	PDM
(Wyatt et al. 2021)	USB 2.0 Mini microphone	-	KISEER	25.4 × 5.08	96	30dB	USB
(Ko et al. 2022)	ReSpeaker 6-Mic Circular Array (MS-M321A3729H-9CP)	MEMS x6	Seeed (Microsystems)	-	59	−22	I2C, PCM
(Strantzalis et al. 2022)	MP34DT01-M	MEMS	STM	3 × 4 × 1.06	61	−26	PDM
(He et al. 2022)	SPH0645LM4H	MEMS	Knowles	3.5 × 2.65 × 0.98	65	−26	I2S
(Somwong et al. 2023)	2xMP34DT05	MEMS x2	STM	3 × 4 × 1	64	−26	PDM
(Marciniak et al. 2023)	MP23ABS1	MEMS	STM	3.5 × 2.65 × 0.98	64	−38	Analog
(Hou et al. 2023)	ICS-43,434	MEMS	TDK	3.5 × 2.65 × 0.98	65	−26	I2S
(Munirathinam and Vitek 2024)	4x SPH0645LM4H	MEMS x4	Knowles	3.5 × 2.65 × 0.98	65	−26d	I2S

vibrations it produces. Further developments, such as those described in (Jung et al. 2022), present a multi-channel piezoelectric acoustic sensor that has the potential to extract different frequency information directly from the media in a manner analogous to that of the human cochlea. Similarly, a graphene strain sensor that could obtain information from the human throat by contact, with a high-sensitivity response, is presented in reference (Wang et al. 2015).

3.3.2 Preprocessing

The preprocessing strategy applied to a given signal depends on the signal characteristics, which are in turn influenced by the specific application. Consequently, Table 3 presents an analysis of various preprocessing strategies and their computational complexity, indexing the sampling frequency and window duration. One key parameter considered in Table 3 is the sampling rate of the signal. Low- and medium-level integration solutions operate with sampling rates up to 48 kHz, covering the full audible range for humans, whereas high-

Table 3 List of preprocessing techniques and hyperparameters used in the analysed studies with their integration level. In STFT parameters W: refers to the window chosen for the FFT algorithm and O: the overlap or shift between successive frames

References	Integration Level	Sampling [kHz]	Window [ms]	Pre-processing algorithm	STFT parameters [ms]	
					W	O
(Montino and Pau 2019)	High	12	500	-	-	-
(Kumari et al. 2019)	Low	48	1000	-	10	5
(Cerutti et al. 2020)	High	16	960	-	25	10
(Naccari et al. 2020)	High	16	1024	PDM bitstream conversion into PCM in hardware. Asymmetric Hanning windowing for FFT. Range clipping [−80.0, 0] dB	64	32
(Wyatt et al. 2021)	Medium	44.1	5000	-	23.2	11.6
(Ko et al. 2022)	Medium	16	40	Amplitude threshold filter to handle silent data	-	-
(Choudhary et al. 2022)	Medium	16	1000	-	25	10
(Strantzalis et al. 2022)	High	16	1024	PDM bitstream conversion PCM in hardware (digital filter for sigma-delta modulators).	64	32
(He et al. 2022)	High	24	30,000	first-order high-pass filter: pre-emphasis filter, Short Time Energy calculation and sliding window analysis to detect the envelope and boundaries of the sound event. Hanning windowing for FFT. Pisewise average filtering on the FFT spectrum to compress each frame by about 97.66%	85.3	-
(Tripathi and Pandey 2023)	Low	44.1–48	1000	Hamming windowing for FFT.	25	10
(Mohaimenuzzaman et al. 2023)	High	20	1510	-	-	-
(Laksono and Prasetio 2023)	Low	16	4000	Trim or padd the audio files to get 4 s of uniform length batch	-	
(Zhang et al. 2023)	Low	16	4000	Hamming windowing for FFT.	64	32
(Süer et al. 2023)	Low	16	400	Noise reduction technique: Spectral Gating, to enhance sound quality	32	16
(Somwong et al. 2023)	High	44.1	1000	-	-	
(Marciniak et al. 2023)	High	16	5000	Noise floor level threshold cropping.	16 or 32	-
(Hammad et al. 2023)	High	0.001	-	Normalization of values between [0,1] using MinMaxScaler function	-	-

Table 3 (continued)

References	Integration Level	Sampling [kHz]	Window [ms]	Pre-processing algorithm	STFT parameters [ms]	
					W	O
(Brighente et al. 2023)	High	16	1024	Noise floor level threshold cropping at −72dB Spectrograms cropped above 300 Hz.	32	16
(Maayah et al. 2023)	High	16	-	-	-	
(Hou et al. 2023)	High	16	1000	Hanning windowing for FFT.	64	32
(Doinea et al. 2024)	High	16	2000	-	20	10
(Shi et al. 2024)	Low	16−8	2000	Pre-emphasis filter high pass filter. Hanning windowing for FFT.	40 or 64 or 400	10 or 32 or 100
(Priebe et al. 2024)	Low	16	3000	-	64	32
(Mou and Milanova 2024)	Medium	16–20	1510	Hanning windowing for FFT.	25	10
(Hyun Choi et al. 2024)	Medium	10	2000	Spectrograms cropped above 512 Hz.	40	10
(Duan et al. 2024)	Low	-	2000	Pre-emphasis filter high pass filter. Normalization of values using MinMaxScaler function. Unknown windowing for FFT.	256 samples	100 samples
(Wißbrock et al. 2024)	Low	16	5000–10,000	Hanning windowing for FFT.	16 or 512	8 or 256
(Wang et al. 2024)	Low	20	3000	Spectrograms cropped below to 2000 Hz.	12.8	6.4
(Libal and Biernacki 2024)	Low	44.1	1000	Unknown windowing for FFT	-	-
(Sammarco et al. 2024)	Low	22.05	500	Long audio files split with a 50% overlap	46.4	23.2
(Munirathinam and Vitek 2024)	High	16	1952	Hamming windowing for FFT.	32	-
(Huang et al. 2024)	High	16	3000	Down sampling to 16 kHz using zero order holder. High pass filter to enhance signal-to-noise ratio. 9th order Butterworth filter with cut-off frequency of 7000 Hz. Signal threshold detection to wake-up the system from power saving mode.	64	16
(Lin et al. 2024)	High	20	1510	Hamming windowing for FFT.	-	-

integration solutions typically use sampling rates of 16 kHz or lower, thereby limiting the frequency range to 8 kHz or less according to Nyquist's theorem. This reduction in sampling frequency can constrain the amount of information that can be extracted from the physical medium. However, it can also help reduce the computational complexity of the solution, as a lower sampling rate results in less data to process, especially when frequency-domain transformations need to be performed.

The ultrasound spectrum range is widely utilized for distance and object detection, proving effective for gesture recognition and indoor localization when combined with ultrasonic sound generators (Bisio et al. 2018). Recently, leveraging the ultrasound range to detect inaudible sounds from various daily-use devices has been shown to enhance indoor localization by generating acoustic 'anchor' points using these sources (Yang et al. 2022b). Furthermore, the study (Cao et al. 2023) demonstrates that the ultrasonic range, up to 192 kHz, can be unlocked on standard smartphones to enable or improve capabilities such as gesture recognition, human presence detection, extended range, and capturing information from AC grid harmonics. This study also demonstrates the effective use of cutting-edge DL algorithms to perform these tasks such as vision transformers. However, the proper integration of these capabilities with DL algorithms is not comprehensively addressed in the literature. A clear technical gap exists between the low sample rates, hardware configurations and processing approaches used in high-integration-level solutions analysed in this study and the high sampling requirements needed to acquire and preprocess ultrasonic data.

Another important consideration presented in Table 3 is the selection of the window length for data processing. The signal must be processed in blocks to ensure that contextually relevant information is provided for time or frequency analysis. The window length is typically chosen based on prior analysis of the data and the phenomena associated with the sound. For example in (Shi et al. 2024) the processes of the steel industry are analysed and it's concluded that the complete occurrence of all processes can be completed in 2 s so the window can be set to this value. In this context, the time window may potentially impact the complexity, memory, and latency of the entire solution.

Further preprocessing strategies could be implemented to reduce the noise from signals or extract specific frequency information from the data as identified in the column *preprocessing algorithm* in Table 3. This could be achieved using pre-emphasis filters, spectral gating, or other DSP techniques, such as Butterworth filters. Normalisation strategies could help the DL algorithms in generalising and being less affected by amplitude information. The MinMaxScaler function, as defined in (Hammad et al. 2023), could be employed for this purpose.

A methodology involving MEMS microphones is presented in (Naccari et al. 2020) (Strantzalis et al. 2022), where the signal transmitted by the microphone is converted from PDM to PCM through the utilisation of a digital filter for sigma-delta modulators, embedded within the hardware the MCU. The DMA peripheral is employed to facilitate real-time windowing and data storage in buffers via hardware. This approach enables the CPU cycles to be allocated to processing and inference, rather than being consumed by the preceding operations.

Given the prevalence of the frequency domain approach in analysing and processing this kind of time-series data, it is necessary to implement a conversion algorithm to facilitate the transition between domains. In most cases, the FFT algorithm is employed, with the STFT being the preferred option (Yang et al. 2019) as it can encode both time and frequency information of the series.

The hyperparameters of these operations and the windowing used for the reduction of the distortion on the FFT are summarized in Table 3 and identified as W for window duration and O for overlapping time between consecutive windows. It can be observed that lower window length values are selected for highly integrated solutions, with a range of 16 to 64ms. Additionally, most applications employ either the Hanning or Hamming window.

3.3.3 Features

The classical approach to sound recognition is based on the extraction of specific features from the audio signal, which depends on the key variables that define the problem to be solved. This approach relies on the perspective and expertise of the solution designers in various signal processing fields. As a result, sound recognition depends on a set of "handcrafted" features that serve as the input for a classification algorithm. The challenge in this approach lies in selecting the optimal set of features, as it is highly dependent on the nature of the sound signal, often making it a trial-and-error process. In (Sharma et al. 2020) an extensive list of different features is presented, categorized into the typical time, frequency, and time-frequency classification domains.

The DL paradigm is essentially the opposite: the input to the classification or detection algorithm is as raw as possible, similar to the original signal. In this approach, the algorithm itself searches for the optimal features within this structure and is fed with the maximum amount of data available. This eliminates the need for handcrafted features and reduces the dependence on the designer's decisions while aiming to improve the overall accuracy and validity of the system. However, it tends to require more computational resources, as the DL algorithm must process a larger volume of data compared to a few "well-defined" features, which can compromise real-time operation in terms of resource consumption. Table 4 presents the different input features from the analysed literature, including the dimensional size of the feature vectors or matrices as an indicator of the computational effort required for their computation and storage.

Most of the solutions evaluated use a hybrid frequency-time input: a spectrogram. The spectrogram displays the variation in frequency content over time, transforming a one-dimensional detection problem into a two-dimensional one, mixing time and frequency information. This method also brings the audio recognition problem closer to image recognition, making it possible to apply well-known computer vision algorithms (Purwins et al. 2019).

It is common practice in the literature to convert the spectrogram to a scale that aligns more closely with human auditory perception, given that human frequency perception is nonlinear the Mel scale is widely used. The Mel scale is typically linear up to 1000 Hz and logarithmic beyond that point, computed through Mel-filter banks (Mukhamediya et al. 2023). This scaling results in the Mel-Spectrogram, and applying a logarithmic scale to the amplitude of the spectrogram gives the Log-Mel-Spectrogram. Any of these representations can be cropped, as shown in (Hyun Choi et al. 2024) or (Wang et al. 2024), allowing for inexpensive computational filtering in the feature extraction step.

The frequency spectrum or time-frequency spectrogram can be transformed into the cepstral domain by computing a logarithmic transformation of the amplitude, followed by an IDFT or DCT (Babaee et al. 2017). This process is used to obtain the cepstral coefficients called MFCCs.

In the work presented in (Wißbrock et al. 2024), several time-frequency and frequency features are analysed and compared as inputs for an anomaly detection model, including Wavelet-Packet Transform, Hilbert-Huang Transform, and other psychoacoustic features.

From an alternative standpoint, some works do not involve any transformation or feature extraction of the signal, allowing the DL algorithm to extract features directly from the raw audio signal. This approach does not enhance or discard any part of the acquired informa-

Table 4 Features and size dimensions chosen for the input of the DL algorithm in the analysed studies with their integration level. In the input dimension column, an (m x n) representation is provided where m is the frequency dimension and n time dimension of the features

References	Integration Level	Input Dimension	Features
(Montino and Pau 2019)	High	(20 × 11)	Mel-frequency cepstral coefficients (MFCC)
(He et al. 2022)	High	(12 × 42)	
(Maayah et al. 2023)	High	-	
(Wißbrock et al. 2024)	Low	Multiple sizes	
(Libal and Biernacki 2024)	Low	(120 × 1)	
(Munirathinam and Vitek 2024)	High	(16 × 61)	
(Lin et al. 2024)	High	(13 × 50)	Mel spectrogram (Mel frequency log-scale power spectrogram)
(Kumari et al. 2019)	Low	(256 × 199)	
(Cerutti et al. 2020)	High	(64 × 96)	
(Naccari et al. 2020)	High	(30 × 32)	
(Wyatt et al. 2021)	Medium	(128 × 430)	
(Strantzalis et al. 2022)	High	(30 × 32)	
(Tripathi and Pandey 2023)	Low	(128 × 100)	
(Zhang et al. 2023)	Low	(40 × 126)	
(Somwong et al. 2023)	High	-	
(Marciniak et al. 2023)	High	(40x-)	
(Brighente et al. 2023)	High	(32 × 62)	
(Hou et al. 2023)	High	(30 × 32)	
(Mou and Milanova 2024)	Medium	(128 × 66)	
(Duan et al. 2024)	Low	-	
(Wißbrock et al. 2024)	Low	Multiple sizes	
(Sammarco et al. 2024)	Low	(128 × 101)	
(Huang et al. 2024)	High	(80 × 184)	
(Lin et al. 2024)	High	(40 × 99)	
(Priebe et al. 2024)	Low	(128 × 128)	Mel spectrogram – Normalized along each frequency bin
(Süer et al. 2023)	Low	(96 × 25)	Magnitude spectrogram
(Doinea et al. 2024)	High	(128 × 200)	
(Lin et al. 2024)	High	(63 × 99)	
(Wißbrock et al. 2024)	Low	Multiple sizes	Wavelet Packet transform
		Multiple sizes	Hilbert Huang Transform
		Multiple sizes	Specific Loudness
		Multiple sizes	Spectral Coefficients
(Libal and Biernacki 2024)	Low	(512 × 1)	Parametric power spectral estimation - Burg Algorithm
		(512 × 1)	Pseudospectrum estimation – MUSIC algorithm
		(120 × 1)	Gammatone cepstral coefficients or GCCCs

Table 4 (continued)

References	Integration Level	Input Dimension	Features
(Hyun Choi et al. 2024)	Medium	(224 × 224)	Magnitude spectrogram + high pass crop above 512 Hz
(Wang et al. 2024)	Low	(110 × 110)	Magnitude spectrogram + high pass crop below 2 kHz
(Shi et al. 2024)	Low	(150 × 150)	MFCC + Chromagram + Wideband Spectrogram + Narrowband Spectrogram
(Ko et al. 2022)	Medium	(1 × 400)	Raw Audio
(Mohaimenuzzaman et al. 2023)	High	(1 × 30225)	
(Laksono and Prasetio 2023)	Low	(1 × 64000)	
(Mou and Milanova 2024)	Medium	(1 × 30225)	
(Huang et al. 2024)	High	(1 × 48000)	
(Hammad et al. 2023)	High	-	Raw Sound Pressure signal
(Choudhary et al. 2022)	Medium	(64 × 96) + (1 × 16000)	Mel spectrogram + Raw Audio

tion, thus giving the algorithm a complete view of the data. However, it tends to require the processing of a larger quantity of data due to the high sample rate needed for audio bandwidth and the fact that frequency representations can compress blocks of time data into a smaller number of features. The final option is a combination of raw waveforms with frequency or time-frequency features, as presented in (Choudhary et al. 2022).

As demonstrated in the analysis synthesized on Table 4, solutions with a high level of integration tend to exhibit lower-dimensional features. This phenomenon facilitates computational resource management but may also result in a less detailed representation of the time or frequency axis, potentially leading to reduced performance in the final task and difficulty in generalizing classification tasks designed for a large number of classes.

In (Mou and Milanova 2024), a comparative analysis is conducted between time-frequency Mel-spectrogram features and raw audio data. The Mel-spectrogram is shown to exhibit superior noise robustness and a reduction in redundancy due to the compression of time information into a lower-dimensional frequency domain. As a result, the Mel-spectrogram is identified as a more efficient representation for processing sound signals, particularly for tasks influenced by human perception. Additionally, the results demonstrate better accuracy on various datasets for classifiers using Mel-spectrograms as input features. Nevertheless, studies such as (Mohaimenuzzaman et al. 2023), (Laksono and Prasetio 2023), (Huang et al. 2024) suggest the potential use of specific DL models designed exclusively for raw data as an input, which could achieve at least the same levels of performance and computational cost.

In terms of computational cost, the authors of (Naccari et al. 2020) present a Log-Mel-Spectrogram with 30 frequency bins or Mel coefficients x 32 time steps (30 × 32) sampled at 16 kHz, computed in an MCU (STM32L476JG) with an execution time of 2.75ms per time step. This results in a total execution time of 88ms for the entire feature matrix. The authors (Wyatt et al. 2021) present the same feature but with a size of 128 × 430 sampled at 44.1 kHz and computed in a Raspberry Pi with a processing time of 575 ms. The comparison in (Huang et al. 2024) demonstrates that the processing time for the computation of a Log-Mel-Spectrogram of 80 × 184 in an MCU (nRF52840) is 1980.259 ms, whereas the processing time for the raw signal is 2 ms. The entire process, including the inference, takes 2386 ms for the first case and 1490 ms for the second case. This demonstrates that, despite the lower inference time required by the time-frequency domain features classification approach, the raw data approach could be, overall, more time-efficient, especially when the spectrogram is large.

3.4 Deep learning algorithms

The core of the applications analysed in this work is the DL algorithms. As defined in (LeCun et al. 2015) these algorithms can uncover intricate structures in high-dimensional data through their multi-layered architecture, making them a preferred solution for designers of classification or pattern detection systems in signal processing.

Table 5 presents the different algorithmic approaches used for the selected tasks in the analysed studies. The analysis also compares the DL approach with the input feature domain approach. As seen, two main tasks can be identified in the sound recognition domain: *the Classification Task and the Anomaly Detection Task*, each with different possible algorithms that can be applied.

3.4.1 Classification task

As outlined in (Deng and Yu 2014), classification task networks are supervised learning algorithms designed to enhance pattern classification through the characterisation of posterior distributions of classes, conditioned on target data. Consequently, a pre-labelled dataset is a prerequisite for training these networks.

As can be observed in Table 5, the most common methodology for classification tasks in sound recognition tasks is the use of CNN algorithms with time-frequency input features or, to a lesser extent, with time features. The CNNs for audio classification are analysed in (Zaman et al. 2023). In summary, CNNs are comprised of multiple convolutional layers with activation functions that produce feature maps from convolution, extracting significant features while introducing non-linear behaviour. Pooling layers are then used to reduce the input size while preserving important information. Finally, fully connected or dense layers handle the classification task and produce, typically, a probability distribution over the possible output classes. The final dense layer may be omitted in FCN, where kernels and shifting kernels with learnable weights are used instead of processing the data all at once, thereby assigning each neuron its weight and bias. This architectural approach minimises the model's parameters and enables the learning of long-range dependencies from audio, as implemented in (Laksono and Prasetio 2023).

As traditional neural networks have the problem of lack of memory, known as gradient vanishing, Recurrent Neural Networks (RNN) use feedback, as loops, that assist in the usage of previously established knowledge (Zelios et al. 2022). This feature enables the network to capture the temporal context of the data, allowing it to recognize patterns in the audio signal over time (Zaman et al. 2023). An LSTM network is a typical form of RNN

Table 5 Input feature domain, deep learning approach and associated task of the analysed studies

References	Input domain approach	DL approach	Task
(Kumari et al. 2019), (Naccari et al. 2020), (Strantzalis et al. 2022), (He et al. 2022), (Zhang et al. 2023), (Somwong et al. 2023), (Marciniak et al. 2023), (Brighente et al. 2023), (Maayah et al. 2023), (Hou et al. 2023), (Doinea et al. 2024), (Shi et al. 2024), (Priebe et al. 2024), (Mou and Milanova 2024), (Hyun Choi et al. 2024), (Duan et al. 2024), (Wang et al. 2024), (Sammarco et al. 2024), (Munirathinam and Vitek 2024), (Huang et al. 2024), (Lin et al. 2024)	Time-Frequency features	CNN	One class or Multiclass classification
(Ko et al. 2022), (Mohaimenuzzaman et al. 2023), (Huang et al. 2024)	Time features	CNN	Multiclass classification
(Montino and Pau 2019), (Cerutti et al. 2020)	Time-Frequency features	CRNN	Multiclass classification
(Laksono and Prasetio 2023)	Time features	FCN	Multiclass classification
(Tripathi and Pandey 2023)	Time-Frequency features	ResNet	Multiclass classification
(Huang et al. 2024)	Time features	Transformer	Multiclass classification
(Wyatt et al. 2021)	Time-Frequency features	Transformer	Multiclass classification
(Mou and Milanova 2024)	Time-Frequency features	LSTM	Multiclass classification
(Choudhary et al. 2022)	Time-Frequency + Time only features	LSTM	Multiclass classification
(Libal and Biernacki 2024)	Frequency features	Autoencoder	Anomaly detection
(Süer et al. 2023)	Time-Frequency features	CNN-based Autoencoder	Anomaly detection
(Hammad et al. 2023)	Time-Features	LSTM-based Autoencoder	Anomaly detection
(Wißbrock et al. 2024)	Time-Frequency features	DNN + GMM, CNN + GMM, Transformer + GMM.	Anomaly detection

that can learn long-term dependencies from sequences of data. An exhaustive analysis and survey of this architecture is presented in (Smagulova and James 2019).

A CRNN is a network comprising several convolutional neural network (CNN) layers, followed by some recurrent neural network (RNN) layers. The combination of CNN and RNN capabilities enhances sound event detection capabilities and adds resilience in complicated sound environments (Xiong et al. 2022). While CNNs are effective at identifying spatial relations and extracting features, RNNs excel at capturing long-term dependencies. In (Xiong et al. 2022), a model that classifies events and identifies the start and end of them using this approach is presented.

The ResNet is based on deep residual learning, designed to address the degradation problem by introducing skip connections or shortcuts between layers, allowing the model to extract more information from the original data (He et al. 2016).

The most recent model analysed is the transformer. A transformer is a transduction model that relies on an attention mechanism to compute representations of its input and output (Nogueira et al. 2022b). Transformer-based methods can handle input length variance due to the multi-head self-attention mechanism, which operates with variable-length input sequences and captures global context information. However, transformers require substantial amounts of data for training (Zaman et al. 2023).

3.4.2 Anomaly detection task

The objective of anomaly detection is to identify states that deviate from the norm within a system, without prior knowledge of the potential anomalous states that may exist within that system. Consequently, no complete data on the whole problem can be obtained. A typical approach involves unsupervised learning to capture the high-order correlation of the observed data when no information about the target labels is available (Deng and Yu 2014).

For this task, the autoencoder represents the fundamental algorithmic construct underlying the solution. As defined in (Zaman et al. 2023), it has two steps: first, an encoder transforms the input data to a lower dimensional representation, and then a decoder recreates the output data from the encoded input.

This architecture is capable of learning the typical behaviour of a system, reproducing it, and generating an error in the decoder output when an anomaly occurs, which can be quantified. To enhance robustness and accuracy, a feature extraction layer could be incorporated into the architecture, using other types of networks, such as CNN or RNN.

Alternative methodologies could be employed to address this task. For instance, in (Wißbrock et al. 2024), a comprehensive representation of the input data is obtained, and then a GMM algorithm is utilised to calculate the similarity of the input to the database. This led to a deep trainless data aggregation approach that could aggregate data to the database without requiring additional training.

3.5 From deep learning to edge computing

3.5.1 Limited resources

Following the structure of the analysis found in (Cerutti et al. 2020), the diverse hardware implementations of the papers under examination are examined in Table 6. The key features compared are NVM that is required to store the weights and biases obtained from the trained model, the RAM that is needed for the buffers that keep the outputs of each layer available during network propagation or inference, the expected power consumption of the device and the achievable by each device, as reported by the manufacturer.

As presented in Table 6 the typical NVM size of an embedded system is defined as around 1024 kB and the RAM memory is between 256 and 512 kB. These resources are not only used by the DL model and its inference but the feature extraction, signal preprocessing and the other functions needed by the device, such as energy monitoring, system monitoring, user interaction, etc.

The computational complexity of the model is typically expressed in various units, including FLOPs, MACCs, and MOPS. However, the MCU computational power is typically referred to in terms of MIPS. A direct conversion between these metrics is not a

Table 6 Comparative of the different processors or boards used to integrate the solutions presented in this review

Device	Manufacturer	Type	NVM [kB]	RAM [kB]	Power [mW]	MIPS
Raspberry Pi Zero	Raspberry	MicroPC	external	4 194 304	500	1 162
Portenta H7	Arduino	MCU	16,384	8 192	1 150	1 024
STM32F767	STM	MCU	2048	512	636.9	462
ESP32 S3	Espressif	MCU	384	512	302.6	288
Sony Spresense	Sony	MCU	8 192	1 536	100	195
Thingy:53	Nordic	MCU	1 024	512	51.2	192
STM32F411	STM	MCU	512	128	33	125
STM32L476	STM	MCU	1 024	128	10.3	100
Nano 33 BLE Sense	Arduino	MCU	1 024	256	11.4	80
STM32WB55	STM	MCU	1 024	256	11.2	80
nRF52840	Nordic	MCU	1 024	256	11.4	80

straightforward process, as a typical operation requires the execution of multiple instructions by a processor. Additionally, the type of operation and the format of the data influence the embedded device’s capabilities. Most 32-bit MCUs support SIMD operations, which allow up to four instructions to be processed in a single clock cycle. This integration allows operations to be performed 2.32 times faster than conventional processing speeds, as shown in (Cerutti et al. 2020) due to SIMD directives and the architectural design of the model. Although the relevance of this type of hardware optimisations or tools is recognised, this is not an extended research point in the studies analysed.

While most studies focus on designing and optimizing DL models, embedded solutions require further consideration. The embedded devices are usually intended to be used in real-time scenarios where the task and power management are critical to the optimal implementation of the overall solution, as in consumer electronics products.

Few studies address real-time operation and typical MCU implementation strategies. For instance, (Somwong et al. 2023) presents a diagram of the embedded classifier subprocess, (Strantzalis et al. 2022) describes the implementation with the microphone using PDM to PCM conversion and DMA to allocate the data directly in the memory of the MCU and preprocess the features in batches, and (Naccari et al. 2020) describes similar process tasks separately. (Maayah et al. 2023) provides a detailed flowchart for the real-time application, while (Lin et al. 2024) offers a comprehensive design for the real-time operation of the entire solution. In (Doinea et al. 2024), the authors adopt an IoT and network perspective, focusing on wireless communication and data flows rather than signal processing and classification tasks. An integration with additional sensors and actuators to implement a global solution is presented in (He et al. 2022). Finally, in (Hou et al. 2023) the authors present a complete solution, analysing it from the points of view of inference computation, preprocessing optimisation and network implementation via mesh structure.

One area that remains underexplored is the integration of deep learning algorithms with low-power strategies, such as those used in standby or sleep modes. These approaches require the implementation of specific algorithms and workflows to enable the partial or total activation and deactivation of the device to conserve energy and extend the operational lifespan of the device. In this case, only (Huang et al. 2024) present a preprocessing algorithm that uses a non-machine learning approach, designed to activate feature extraction and inference algorithms exclusively when a potential event is detected, rather than in response to silence or an absence of sufficient sound.

3.5.2 Optimization and compression techniques

As analysed in (Liu et al. 2019), the most advanced DL models for a range of tasks have a multitude of parameters, spanning from 1 M to 134 M, and exhibit a considerable computational complexity, spanning from 23 M to 4G FLOPs. These models are typically characterized by a floating-point representation of weights and biases and use intricate operations, such as discrete convolutions, which cannot be efficiently embedded for large-scale calculations on a MicroPC or smartphone, let alone directly into an MCU. Their computational requirements are orders of magnitude beyond what an MCU can handle (see Table 6), requiring the use of lightweight architectures and various optimization strategies to fully embed DL algorithms.

Table 7 presents a comparison between the analysed models, including the complexity of the objective task, the optimization strategies, the results accuracy and the model size.

Regarding the possible comparison parameters, it is evident that there is a lack of clarity and standardisation in the metric framework employed in the analysed literature. The specific performance metrics employed vary depending on the nature of the work in question. Such metrics may include, for instance, a percentage accuracy or the area under the curve (AUC), contingent on the distribution of the dataset or the author's preference. Moreover, the way model complexity is quantified, and the presentation of memory size may vary depending on whether the solution is fully integrated in an embedded device. In such a case, the values of NVM and RAM, or the inference time in the device, are typically known.

Most of the papers examined begin with a model that has a reduced structure compared to state-of-the-art models typically run on computers. In (Kumari et al. 2019) for example, the L^3-audio network, which has an approximate size of 18 MB, was pruned down to a size of 0.814 MB, representing a reduction of 95%. This resulted in a loss of 1.4% in performance. The model pruning, also known as sparsity, is a very used technique to reduce the size of a model. This technique sets a predefined number of weights to zero during the training, and then the operations with the zero weights are removed from the model (Vandendriessche et al. 2021). In (Mohaimenuzzaman et al. 2023), for instance, the CNN channel pruning technique is developed using magnitude-based and Taylor criteria-based ranking, resulting in a reduction of up to 97% in the size of the network while maintaining an accuracy level of 83.65%.

This kind of reduced models are known as lightweight models and, as introduced in (Saha et al. 2022), the integration of DL models in edge devices usually starts with the choosing of a model from the lightweight *zoo* based on the application and hardware specifications. The DL algorithm column in Table 7 can be considered the model *zoo* of the last publications.

In complex tasks, the lightweight models may not provide sufficient performance. In such instances, KD can be employed to transfer the learning characteristics learned by a *cumbersome* model to a lightweight one (Hinton et al. 2015). A comprehensive survey of various KD methodologies is presented in (Tripathi and Pandey 2023) which also examines an application of KD where knowledge is transferred from several teacher networks to a ResNet.

Another approach, proposed in (Duan et al. 2024), utilizes structural re-parameterization techniques, such as merging convolution layers and cascading activation functions, to decouple the training structure from the inference structure. This methodology enables independent optimization of training and inference stages while ensuring that the model captures the appropriate feature information during the training phase, specifically for a 20-class classification problem.

The typical precision for the weights and activations of a neuronal network is a 32-bit floating point, therefore an 8-bit quantization of these variables is a highly effective technique for optimizing the models (Mou and Milanova 2024). Quantization reduces the overall size of the model and significantly speeds up inference times, as 8-bit integer operations are computationally less expensive than floating-point operations (Mou and Milanova 2024). In (Naccari et al. 2020), a comparison between an unquantized model and an 8-bit quantized version showed a 44% reduction in inference time, a 28% decrease in RAM usage, and a 25% reduction in NVM usage, with a slight accuracy loss of 1.1%.

The quantization rate could overcome accuracy problems in certain applications. In (Cerutti et al. 2020) accuracy of the model is set as the representative metric to design a quantization framework with out-of-range provability and signal-to-qualization-noise-ratio metrics. Even in (Sammarco et al. 2024), a hybrid quantization approach is presented with an 8-bit integer for weights but an original representation in floating-point for biases and activations. In (Novac et al. 2021) a quantization framework is presented for the deployment of DL models onto low-power 32-bit MCUs. In the survey presented in (Li et al. 2023), the different quantization approaches are analysed, among other compression techniques. As evidenced in Table 7, most high-integration solutions have opted for quantization, particularly post-training quantization.

Furthermore, high-integration solutions often utilize pre-designed frameworks to simplify model deployment. These include frameworks such as TensorFlow lite or TFlite, STM32CubeAi or Edge Impulse. These frameworks provide common strategies to train and optimize DL models for edge applications. A comprehensive analysis of these frameworks is provided in (Saha et al. 2022).

Additional optimizations can be considered for further improving efficiency. For instance, in (Brighente et al. 2023) the use of ReLU as an activation function is recommended due to its straightforward integration into MCUs, eliminating the need for lookup tables typically required for other nonlinear functions, such as the hyperbolic tangent. To reduce computational effort, a one-dimensional convolution with depth-wise convolution as a convolution factorising method is employed in (Ko et al. 2022). Also in the same work, the use of the TVM compiler (Chen et al. 2018) reduced the inference time. Additionally, partial convolution with sliding windows and iterative point-by-point computation on the output channels of the final convolution layer with average pooling can significantly reduce memory consumption during these computations, as shown in (Huang et al. 2024).

4 Conclusion and future work

The selected papers of this review are analysed in Tables 1, 3, 4 and 7. Of the selected papers, 44% focus on highly integrated solutions in embedded systems using MCUs, 41% resent solutions integrated into portable devices such as smartphones or microPCs, and 15%

Table 7 DL approach and associated compression techniques to address the edge paradigm. Comparative of results , size and complexity for the solutions with their integration level

References	Integration Level	Task*	DL Algorithm	Compression techniques	Accuracy	Size			Complexity	
						Params	NVM [kB]	RAM [kB]	Ops	Inference time [ms]
(Montino and Pau 2019)	High	MC-3C	CRNN (w/ LSTM)	-Lightweight model. - STM32Cube.AI framework	95%	2300	17	9	86,465 MACCs	12.8
(Kumari et al. 2019)	Low	MC-50 C	EdgeL^3 (CNN)	-Magnitude base pruning. -fine-grained pruning for individual weights. -Coarse-grained dropping for entire filters or layers. -FT and KD to compensate for loss in performance.	72.25%	213,491	833.5 (SP) 416.7 (HP)	-	-	-
(Cerutti et al. 2020)	High	MC-10 C	CRNN (VGGish as extractor w/ GRU as classifier)	−2 steps KD: from VGGish to student 1, to student 1 to student 2. −8-bit quantization (post-training) using CMSIS-NN. -Use of assembly directives MAC and SIMD.	72.67%	30,600	30.6	34.3	2.11 MOPS	125
(Naccari et al. 2020)	High	MC-3C	CNN	-Lightweight model. -STM32Cube.AI framework. −8-bit quantization (post-training).	89.17%	7867	7.71	4.94	-	36.02
(Wyatt et al. 2021)	Medium	MC-6C	Tiny BERT-based transformer	-Lightweight model. (mapping layer to reduce size). -TFLite framework.	81.2%	14,850	78.4	-	-	379
(Ko et al. 2022)	Medium	MC-12 C	CNN (1D)	-Use of efficient multi-stream block for 1D convolutions (depth-wise separable convolution). -Use of TVM compiler.	>90%	-	-	-	-	7.811

Table 7 (continued)

References	Integration Level	Task*	DL Algorithm	Compression techniques	Accuracy	Size			Complexity	
						Params	NVM [kB]	RAM [kB]	Ops	Inference time [ms]
(Choudhary et al. 2022)	Medium	MC-200 C	LEAN Wave Encoder (Bidirectional LSTM-based) and logmel-based pretrained YAMnet with attention-based temporal alignment.	-TFLite framework. -Undefined quantization.	0.944 (mAUC)	4,580,000	4608	-	-	65
(Strantzalis et al. 2022)	High	MC-3C	CNN (2D or 1D kernels)	-Lightweight models. -TFLite framework + STM32Cube. AI framework or Edge Impulse framework −8-bit quantization (post-training).	93.3% or 97.8%	-	7.8 or 38.2	5.52 or 7	501,428 MACCs or unknown	40.76 or 16
(He et al. 2022)	High	MC-2C	CNN	-Lightweight model. -TFLite framework + STM32Cube. AI framework or Edge Impulse framework −8-bit quantization (post-training).	99.25%	-	91.54	12.61	-	-
(Tripathi and Pandey 2023)	Low	MC-10 C**	ResNet-18	-Lightweight model. -Trained with cumbersome models using different KD techniques.	93.97%	11,700,000	-	-	7.95 GMACCs	
(Mohaimenuzzaman et al. 2023)	High	MC-50 C	Micro-ACDNet (CNN)	-Lightweight model. -Hybrid (channel and weights) pruning: magnitude or Taylor criteria-based ranking. -TFLite Micro framework −8-bit quantization (post-training).	83.65% or 71% (with quantization)	131,000	500	303 or 153	21.5 MFLOPs	-
(Laksono and Prasetio 2023)	Low	MC-30 C MC-125 C	Fully Convolutional Quartznet (FCN)	-Lightweight model due to Time Channel Separable Convolution Architecture.	84.6% or 56.4%	53,000	-	-	-	-

Table 7 (continued)

References	Integration Level	Task*	DL Algorithm	Compression techniques	Accuracy	Size			Complexity	
						Params	NVM [kB]	RAM [kB]	Ops	Inference time [ms]
(Zhang et al. 2023)	Low	MC-7C	VGG-like model (CNN)	-lightweight model. -Adaptative pooling	>0.85(mAUC)	75,000	-	-	10 MMACs	-
(Süer et al. 2023)	Low	AD	CNNAE-FT (CNN-based autoencoder)	-Not specified	0.85(mAUC)	-	-	-	-	-
(Somwong et al. 2023)	High	MC-4C	CNN (1D)	-Edge Impulse framework with unspecified parameters.	93.7%	-	-	-	-	-
(Marciniak et al. 2023)	High	MC-6C	CNN	-Edge Impulse framework with unspecified parameters.	81.8%	-	29.2	10.5	-	11–15
(Hammad et al. 2023)	High	AD	LSTM Autoencoder	-Lightweight model. -TFLite framework. -Undefined quantization.	99.34%	-	800.63	102.62	-	4
(Brighente et al. 2023)	High	MC-2C	CNN	-Lightweight model. -TFLite framework. −8-bit quantization (post-training).	97.84%	-	61.9	50.0	-	332
(Maayah et al. 2023)	High	MC-4C	CNN (1D)	-Lightweight model. -TFLite framework. −8-bit quantization (post-training).	85.89%		45.2	7.7	-	1
(Hou et al. 2023)	High	MC-3C	CNN	-Lightweight model. - STM32Cube.AI framework. −8-bit quantization (post-training).	92.5%	-	4.93	3.58	159,606 MACCs	-
(Doinea et al. 2024)	High	MC-2C	CNN (1D)	-Edge Impulse framework −8-bit quantization (post-training). -EON Compiler and TFLite	98.3%	-	-	28	-	51
(Shi et al. 2024)	Low	MC-7C	M-VGG	-Lightweight model. - Stochastic pooling. Global average pooling.	91.26%	14,720,000	-	-	-	-

Table 7 (continued)

References	Integration Level	Task*	DL Algorithm	Compression techniques	Accuracy	Size			Complexity	
						Params	NVM [kB]	RAM [kB]	Ops	Inference time [ms]
(Priebe et al. 2024)	Low	OC	MobileNetV3-Small-Pi	-KD techniques: soft target distillation, feature-based distillation and relational-based distillation from EcoVAD Teacher.	0.986(mAUC)	52,253	194.56	-	4,324,170 MACCs	5
(Mou and Milanova 2024)	Medium	MC-50 C	CNN, LSTM	-Specific pruning process with magnitude, Taylor and hybrid pruning. −8-bit quantization	85.25%	4,735,378	157	-	544.4 MFLOPs	2.12
(Hyun Choi et al. 2024)	Medium	MC-2C	EfficientNetV2B0-like model. (CNN)	-Not specified	95.6%	-	-	-	-	-
(Duan et al. 2024)	Low	MC-20 C	SIAlex (CNN)	-Lightweight model: - Merging of convolution layers - Cascading activation functions. -Decoupling the training structure from the inference structure.	93.66%	-	-	-	-	-
(Wißbrock et al. 2024)	Low	AD	-DenseNet21 -MobileViTXXS -Openl3EnvLinear -musiccnnMTT + GMM for anomaly detection	-Lightweight models	0.732–0.925 (mAUC)	600,000–7,000,000	-	-	-	-
(Wang et al. 2024)	Low	MC-2C	Light-CNN	-Lightweight model.	100%	201,378	2426.9	-	-	-
(Libal and Biernacki 2024)	Low	AD	Autoencoder+MSE threshold classifier	-Lightweight model.	99.66–99.97%	-	-	-	-	-

Table 7 (continued)

References	Integration Level	Task*	DL Algorithm	Compression techniques	Accuracy	Size			Complexity	
						Params	NVM [kB]	RAM [kB]	Ops	Inference time [ms]
(Sammarco et al. 2024)	Low	MC-7C	MobileNetV2 (CNN)	-TFLite framework. - Float 16 quantization, or - Hybrid quantization: 8-bit integer for weights but biases and activations with original representation.	0.904(MCC***) or 0.72(MCC***)	-	5120 or 2560	-	-	-
(Munirathinam and Vitek 2024)	High	OC	CNN	-Lightweight models. -TFLite framework- - STM32Cube.AI framework. −32-bit quantization (post-training).	90%	-	-	-	-	120
(Huang et al. 2024)	High	MC-2C	CNN 2D, CNN 1D or Transformer	-Lightweight models. −8-bit quantization (post-training). -Partial convolution technique and average pooling. -Baseline filtering technique to make a preliminary analysis and wake-up MCU.	99%, 94% or 93%	-	104.32, 75.56 or 83.46	37.87, 24.10 or 24.7	-	406.1, 1490.69 or 1079.29
(Lin et al. 2024)	High	MC-2C	CNN (1D)	-Lightweight model. -TFLite Micro framework. −8-bit quantization (post-training).	81.25%	-	34.7	16.8	-	1

*OC: One-Class Classification. MC-XC: Multi-Class Classification, where X denotes the number of classes. AD: Anomaly Detection

**Generalization within different datasets

***Matthews Correlation Coefficient (MCC) describe solutions not directly integrated into devices, but which offer interesting techniques for the future of the field.

Given the significant differences between running code on a MicroPC versus an MCU—such as in perspective, design, and the tools involved—it is challenging to compare these approaches directly. Nonetheless, a clear distinction exists in the complexity of the tasks: more complex classification or anomaly detection tasks are typically run on MicroPCs, rather than MCUs. This highlights the need to develop further optimization techniques and advances in MCU hardware technologies to enable the development of integrated electronic products with DL-based sound recognition.

One challenge when presenting research findings is the lack of a standardized framework that offers a complete solution perspective. This makes it difficult to compare results and accurately understand the improvements and challenges in this field. While it is understandable that metrics vary depending on the chosen problem, dataset characteristics, DL model architecture, and development framework, greater efforts should be made to present results in terms of computational complexity, memory usage, and task performance.

Regarding signal acquisition, MEMS technology is widely used, but further research is needed on its suitability in specific situations—such as noisy environments—along with other relevant considerations such as power consumption and frequency range.

The preprocessing stages and feature extraction from sound signals are often underexplored, especially considering their importance. Some studies neglect to address issues like windowing in FFT, the length of the sound data, or the sampling rate during digital acquisition. Furthermore, many solutions could benefit from filtering systems, including hardware-based filters, to improve performance.

Audio signals exhibit useful information content across a broad frequency range, in some cases extending beyond the limits of human perception. Low integrated solutions typically offer high sampling rates, capturing higher frequency information. However, for highly integrated designs, the sample rate emerges as a critical parameter significantly impacting resource demands. Consequently, solutions embedded in MCUs often utilize the minimal possible frequency content. In this context, a promising research direction lies in investigating optimization or modulation techniques that enable resource-constrained devices to acquire and process the ultrasound portion of the spectrum.

The literature presents an extensive zoo of lightweight models and possibilities, but the most advanced and recent ones are not yet ready for a straightforward implementation to the very edge devices; from this point of view, further research could be done applying new techniques to well-known problems or even specific applications, rather than relying on the standard Mel-spectrogram and CNN configurations that dominate the current approaches.

There is a clear lack of information on applying DL approaches in real-time scenarios with the constraints of real-time devices. These devices often share computational resources for functions such as user interaction, device-specific operations, the operating system, and power management. From this perspective, further research should focus on evaluating the real-world integration of the proposed algorithms and solutions into end products. Task scheduling, efficient use of hardware peripherals, optimized feature extraction, energy-saving modes, and other non-DL strategies should be explored in combination to ensure an optimal balance of system performance and resource management.

Acknowledgements This work was supported in part by the 'Ajuts UdL-Banc Santander per a la contractació de personal Predoctoral en Formació'.

Author contributions P.G. conceived the review, developed the methodology, analysed the literature, designed the figures, and wrote the initial draft. T.P. contributed to critical revisions of the manuscript, wrote sections of the manuscript, and reviewed the figures. M.T. supervised the project. All authors reviewed and approved the final manuscript.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

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

Declarations

Competing interests The authors declare no competing interests.

References

Alsina-Pagès RM, Hervás M, Duboc L, Carbassa J (2020) Design of a low-cost configurable acoustic sensor for the rapid development of sound recognition applications. Electronics 9:1155. https://doi.org/10.3390/electronics9071155

Babaee E, Anuar NB, Abdul Wahab AW, Shamshirband S, Chronopoulos AT (2017) An overview of audio event detection methods from feature extraction to classification. Appl Artif Intell 31:661–714. https://doi.org/10.1080/08839514.2018.1430469

Bahai A (2024) Making sense at the edge. In: 2024 IEEE symposium on VLSI technology and circuits (VLSI Technology and Circuits). pp 1–2. https://doi.org/10.1109/VLSITECHNOLOGYANDCIR46783.2024.10631378

Bansal A, Garg NK (2022) Environmental sound classification: a descriptive review of the literature. Intell Syst Appl 16:200115. https://doi.org/10.1016/j.iswa.2022.200115

Bisio I, Delfino A, Grattarola A, Lavagetto F, Sciarrone A (2018) Ultrasounds-based context sensing method and applications over the internet of things. IEEE Internet Things J 5:3876–3890. https://doi.org/10.1109/JIOT.2018.2845099

Branding J, von Hörsten D, Böckmann E, Wegener JK, Hartung E (2024) InsectSound1000 an insect sound dataset for deep learning based acoustic insect recognition. Sci Data 11:475. https://doi.org/10.1038/s41597-024-03301-4

Brighente A, Conti M, Peruzzi G, Pozzebon A (2023) ADASS: anti-drone audio surveillance sentinel via embedded machine learning. In: 2023 IEEE sensors applications symposium (SAS). pp 1–6. https://doi.org/10.1109/SAS58821.2023.10254008

Canziani A, Paszke A, Culurciello E (2017) An Analysis of Deep Neural Network Models for Practical Applications. ArXiv. https://doi.org/10.48550/arXiv.1605.07678

Cao S, Li D, Lee SI, Xiong J (2023) PowerPhone: unleashing the acoustic sensing capability of smartphones. In: Proceedings of the 29th annual international conference on mobile computing and networking. Association for Computing Machinery, New York, pp 1–16. https://doi.org/10.1145/3570361.3613270

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 Sel Top Signal Process 14:654–664. https://doi.org/10.1109/JSTSP.2020.2969775

Chachada S, Kuo C-CJ (2014) Environmental sound recognition: a survey. APSIPA Trans Signal Inf Process 3:e14. https://doi.org/10.1017/ATSIP.2014.12

Chandrakala S, Jayalakshmi SL (2019) Environmental audio scene and sound event recognition for autonomous surveillance: a survey and comparative studies. ACM Comput Surv 52:63:1–6334. https://doi.org/10.1145/3322240

Chen T, Moreau T, Jiang Z, Zheng L, Yan E, Cowan M, Shen H, Wang L, Hu Y, Ceze L, Guestrin C, Krishnamurthy A (2018) TVM: an automated end- to-end optimizing compiler for deep learning. ArXiv. https://doi.org/10.48550/arXiv.1802.04799

Choudhary S, Karthik CR, Lakshmi PS, Kumar S (2022) LEAN: light and efficient audio classification network. In: 2022 IEEE 19th India council international conference (INDICON). pp 1–6. https://doi.org/10.1109/INDICON56171.2022.10039921

Ciaburro G, Iannace G (2020) Improving Smart cities Safety using sound events detection based on deep neural network algorithms. Informatics 7:23. https://doi.org/10.3390/informatics7030023

Deng L, Yu D (2014) Deep learning: methods and applications. Found Trends® Signal Process 7:197–387. https://doi.org/10.1561/2000000039

Doinea M, Trandafir I, Toma C-V, Popa M, Zamfiroiu A (2024) IoT embedded smart monitoring system with edge machine learning for beehive management. Int J Comput Commun Control. https://doi.org/10.15837/ijccc.2024.4.6632

Duan L, Yang L, Guo Y (2024) SIAlex: species identification and monitoring based on bird sound features. Ecol Inf 81:102637. https://doi.org/10.1016/j.ecoinf.2024.102637

Hammad SS, Iskandaryan D, Trilles S (2023) An unsupervised TinyML approach applied to the detection of urban noise anomalies under the smart cities environment. Internet Things 23:100848. https://doi.org/10.1016/j.iot.2023.100848

Han X, Peng J (2024) Bird sound detection based on sub-band features and the perceptron model. Appl Acoust 217:109833. https://doi.org/10.1016/j.apacoust.2023.109833

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp 770–778. https://doi.org/10.1109/CVPR.2016.90

He C, Tan J, Jian X, Zhong G, Wu H, Cheng L, Lin J (2022) A novel snore detection and suppression method for a flexible patch with MEMS microphone and accelerometer. IEEE Internet Things J 9:25791–25804. https://doi.org/10.1109/JIOT.2022.3199085

Hinton G, Vinyals O, Dean J (2015) Distilling the knowledge in a neural network. ArXiv. https://doi.org/10.48550/arXiv.1503.02531

Hou L, Duan W, Xuan G, Xiao S, Li Y, Li Y, Zhao J (2023) Intelligent microsystem for sound event recognition in edge computing using end-to-end mesh networking. Sensors 23:3630. https://doi.org/10.3390/s23073630

Huang Z, Tousnakhoff A, Kozyr P, Rehausen R, Bießmann F, Lachlan R, Adjih C, Baccelli E (2024) TinyChirp: bird song recognition using TinyML models on low-power wireless acoustic sensors. In: 2024 IEEE 5th international symposium on the internet of sounds (IS2). pp 1–10. https://doi.org/10.1109/IS262782.2024.10704131

Hyun Choi D, Ha Joo Y, Hong Kim K, Ho Park J, Joo H, Kong H-J, Lee H, Jun Song K, Kim S (2024) A development of a sound recognition-based cardiopulmonary resuscitation training system. IEEE J Transl Eng Health Med 12:550–557. https://doi.org/10.1109/JTEHM.2024.3433448

Jung YH, Pham TX, Issa D, Wang HS, Lee JH, Chung M, Lee B-Y, Kim G, Yoo CD, Lee KJ (2022) Deep learning-based noise robust flexible piezoelectric acoustic sensors for speech processing. Nano Energy 101:107610. https://doi.org/10.1016/j.nanoen.2022.107610

Kim J, Lee H, Jeong S, Ahn S-H (2021) Sound-based remote real-time multi-device operational monitoring system using a convolutional neural network (CNN). J Manuf Syst 58:431–441. https://doi.org/10.1016/j.jmsy.2020.12.020

Ko J, Kim H, Kim J (2022) Real-time sound source localization for low-power IoT devices based on Multi-stream CNN. Sensors 22:4650. https://doi.org/10.3390/s22124650

Küçüktopcu O, Masazade E, Ünsalan C, Varshney PK (2019) A real-time bird sound recognition system using a low-cost microcontroller. Appl Acoust 148:194–201. https://doi.org/10.1016/j.apacoust.2018.12.028

Kumari S, Roy D, Cartwright M, Bello JP, Arora A (2019) EdgeL^3: compressing L^3-Net for mote scale urban noise monitoring. In: 2019 IEEE international parallel and distributed processing symposium workshops (IPDPSW). pp 877–884. https://doi.org/10.1109/IPDPSW.2019.00145

Laksono BSP, Prasetio BH (2023) Speaker recognition on low power device using fully convolutional QuartzNet. In: Proceedings of the 8th international conference on sustainable information engineering and technology. Association for Computing Machinery, New York, pp 619–624. https://doi.org/10.1145/3626641.3626946

LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444. https://doi.org/10.1038/nature14539

Li J, Dai W, Metze F, Qu S, Das S (2017) A comparison of deep learning methods for environmental sound detection. In: 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP). pp 126–130. https://doi.org/10.1109/ICASSP.2017.7952131

Li Z, Li H, Meng L (2023) Model compression for deep neural networks: a survey. Computers 12:60. https://doi.org/10.3390/computers12030060

Libal U, Biernacki P (2024) Non-intrusive system for honeybee recognition based on audio signals and maximum likelihood classification by autoencoder. Sensors 24:5389. https://doi.org/10.3390/s24165389

Lin T-H, Chang C-T, Zhuang T-H, Putranto A (2024) Real-time hollow defect detection in tiles using on-device tiny machine learning. Meas Sci Technol 35:056006. https://doi.org/10.1088/1361-6501/ad2665

Liu J, Liu J, Du W, Li D (2019) Performance analysis and characterization of training deep learning models on mobile device. In: 2019 IEEE 25th international conference on parallel and distributed systems (ICPADS). pp 506–515. https://doi.org/10.1109/ICPADS47876.2019.00077

Maayah M, Abunada A, Al-Janahi K, Ahmed ME, Qadir J (2023) LimitAccess: on-device TinyML based robust speech recognition and age classification. Discov Artif Intell 3:8. https://doi.org/10.1007/s44163-023-00051-x

Marciniak F, Marciniak W, Marciniak T (2023) Analysis of fast prototyping of microcontroller-based ML software for acoustic signal classification. In: 2023 Signal processing: algorithms, architectures, arrangements, and applications (SPA). pp 36–41. https://doi.org/10.23919/SPA59660.2023.10274443

Meedeniya D, Ariyarathne I, Bandara M, Jayasundara R, Perera C (2023) A survey on deep learning based forest environment sound classification at the edge. ACM Comput Surv 56:1–6636. https://doi.org/10.1145/3618104

Mohaimenuzzaman M, Bergmeir C, Meyer B (2022) Pruning vs XNOR-net: a comprehensive study of deep learning for audio classification on edge-devices. IEEE Access 10:6696–6707. https://doi.org/10.1109/ACCESS.2022.3140807

Mohaimenuzzaman M, Bergmeir C, West IT, Meyer B (2023) Environmental sound classification on the edge: a pipeline for deep acoustic networks on extremely resource-constrained devices. Pattern Recognit 133:109025. https://doi.org/10.1016/j.patcog.2022.109025

Mohammad A, Tripathi DMM (2019) Audio analysis and classification: a review. Int J Res Advent Technol 7:103–109. https://doi.org/10.32622/ijrat.76201926

Montino P, Pau D (2019) Environmental intelligence for embedded real-time traffic sound classification. In: 2019 IEEE 5th international forum on research and technology for society and industry (RTSI). pp 45–50. https://doi.org/10.1109/RTSI.2019.8895517

Mou A, Milanova M (2024) Performance analysis of deep learning model-compression techniques for audio classification on edge devices. Science 6:21. https://doi.org/10.3390/sci6020021

Mukhamediya A, Fazli S, Zollanvari A (2023) On the effect of log-mel spectrogram parameter tuning for deep learning-based speech emotion recognition. IEEE Access 11:61950–61957. https://doi.org/10.1109/ACCESS.2023.3287093

Munirathinam R, Vitek S (2024) Sound source localization and classification for emergency vehicle siren detection using resource constrained systems. In: 2024 34th International conference radioelektronika (RADIOELEKTRONIKA). pp 1–5. https://doi.org/10.1109/RADIOELEKTRONIKA61599.2024.10524053

Naccari F, Guarneri I, Curti S, Savi AA (2020) Embedded acoustic scene classification for low power microcontroller devices. In: Detection and classification of acoustic scenes and events, DCASE2020. Tokyo, Japan.

Nogueira AFR, Oliveira HS, Machado JJM, Tavares JMRS (2022a) Sound classification and processing of urban environments: a systematic literature review. Sensors 22:8608. https://doi.org/10.3390/s22228608

Nogueira AFR, Oliveira HS, Machado JJM, Tavares JMRS (2022b) Transformers for urban sound classification—a comprehensive performance evaluation. Sensors 22:8874. https://doi.org/10.3390/s22228874

Novac P-E, Boukli Hacene G, Pegatoquet A, Miramond B, Gripon V (2021) Quantization and deployment of deep neural networks on microcontrollers. Sensors 21:2984. https://doi.org/10.3390/s21092984

Polo-Rodriguez A, Vilchez Chiachio JM, Paggetti C, Medina-Quero J (2021) Ambient sound Recognition of Daily Events by means of Convolutional Neural Networks and fuzzy temporal restrictions. Appl Sci 11:6978. https://doi.org/10.3390/app11156978

Prashanth A, Jayalakshmi SL, Vedhapriyavadhana R (2024) A review of deep learning techniques in audio event recognition (AER) applications. Multimed Tools Appl 83:8129–8143. https://doi.org/10.1007/s11042-023-15891-z

Priebe D, Ghani B, Stowell D (2024) Efficient speech detection in environmental audio using acoustic recognition and knowledge distillation. Sensors 24:2046. https://doi.org/10.3390/s24072046

Purwins H, Li B, Virtanen T, Schlüter J, Chang S, Sainath T (2019) Deep learning for audio signal processing. IEEE J Sel Top Signal Process 13:206–219. https://doi.org/10.1109/JSTSP.2019.2908700

Qurthobi A, Maskeliūnas R, Damaševičius R (2022) Detection of mechanical failures in Industrial machines using overlapping acoustic anomalies: a systematic literature review. Sensors 22:3888. https://doi.org/10.3390/s22103888

Saha SS, Sandha SS, Srivastava M (2022) Machine learning for microcontroller-class hardware: a review. IEEE Sens J 22:21362–21390. https://doi.org/10.1109/JSEN.2022.3210773

Sammarco M, Stellantis TZ, Gantert L, Campista MEM (2024) Sound event detection via pervasive devices for mobility surveillance in smart cities. In: 2024 IEEE international conference on pervasive computing and communications workshops and other affiliated events (PerCom Workshops). pp 581–586. https://doi.org/10.1109/PerComWorkshops59983.2024.10503381

Sharma G, Umapathy K, Krishnan S (2020) Trends in audio signal feature extraction methods. Appl Acoust 158:107020. https://doi.org/10.1016/j.apacoust.2019.107020

Shi R, Zhang F, Li Y (2024) Lightweight network based features fusion for steel rolling ambient sound classification. Eng Appl Artif Intell 133:108382. https://doi.org/10.1016/j.engappai.2024.108382

Smagulova K, James AP (2019) A survey on LSTM memristive neural network architectures and applications. Eur Phys J Spec Top 228:2313–2324. https://doi.org/10.1140/epjst/e2019-900046-x

Somwong B, Kumphet K, Massagram W (2023) Acoustic monitoring system with ai threat detection system for forest protection. In: 2023 20th International joint conference on computer science and software engineering (JCSSE). pp 253–257. https://doi.org/10.1109/JCSSE58229.2023.10202043

Strantzalis K, Gioulekas F, Katsaros P, Symeonidis A (2022) Operational state recognition of a DC motor using edge artificial intelligence. Sensors 22:9658. https://doi.org/10.3390/s22249658

Süer S, Köseoğlu İ, Öner R, İnce G (2023) Detection of clips failures in manufacturing using audio signals. In: 2023 5th International congress on human-computer interaction, optimization and robotic applications (HORA). pp 01–05. https://doi.org/10.1109/HORA58378.2023.10156765

Svrzić S, Djurković M, Vukićević A, Nikolić Z, Mihailović V, Dedić A (2024) Sound classification and power consumption to sound intensity relation as a tool for wood machining monitoring. Eur J Wood Wood Prod. https://doi.org/10.1007/s00107-024-02139-2

Tang L, Tian H, Huang H, Shi S, Ji Q (2023) A survey of mechanical fault diagnosis based on audio signal analysis. Measurement 220:113294. https://doi.org/10.1016/j.measurement.2023.113294

Tolstikhin I, Houlsby N, Kolesnikov A, Beyer L, Zhai X, Unterthiner T, Yung J, Steiner A, Keysers D, Uszkoreit J, Lucic M, Dosovitskiy A (2021) MLP-mixer: an all-MLP architecture for vision. In: arXiv.org. https://arxiv.org/abs/2105.01601v4. Accessed 23 Jul 2024

Tripathi AM, Pandey OJ (2023) Divide and distill: new outlooks on knowledge distillation for environmental sound classification. IEEEACM Trans Audio Speech Lang Process 31:1100–1113. https://doi.org/10.1109/TASLP.2023.3244507

Turchet L, Fazekas G, Lagrange M, Ghadikolaei HS, Fischione C (2020) The internet of audio things: state of the art, vision, and challenges. IEEE Internet Things J 7:10233–10249. https://doi.org/10.1109/JIOT.2020.2997047

Vandendriessche J, Wouters N, da Silva B, Lamrini M, Chkouri MY, Touhafi A (2021) Environmental sound recognition on embedded systems: from FPGAs to TPUs. Electronics 10:2622. https://doi.org/10.3390/electronics10212622

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2023) Attention is all you need. In: NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems. pp 6000-6010.

Wang Y, Yang T, Lao J, Zhang R, Zhang Y, Zhu M, Li X, Zang X, Wang K, Yu W, Jin H, Wang L, Zhu H (2015) Ultra-sensitive graphene strain sensor for sound signal acquisition and recognition. Nano Res 8:1627–1636. https://doi.org/10.1007/s12274-014-0652-3

Wang J-J, Sharma AK, Liu S-H, Zhang H, Chen W, Lee T-L (2024) Prediction of vascular access stenosis by lightweight convolutional neural network using blood flow sound signals. Sensors 24:5922. https://doi.org/10.3390/s24185922

Wißbrock P, Ren Z, Pelkmann D (2024) More than spectrograms: deep representation learning for machinery fault detection. Appl Acoust 225:110178. https://doi.org/10.1016/j.apacoust.2024.110178

Wyatt S, Elliott D, Aravamudan A, Otero CE, Otero LD, Anagnostopoulos GC, Smith AO, Peter AM, Jones W, Leung S, Lam E (2021) Environmental sound classification with tiny transformers in noisy edge environments. In: 2021 IEEE 7th world forum on internet of things (WF-IoT). pp 309–314. https://doi.org/10.1109/WF-IoT51360.2021.9596007

Xiong W, Xu X, Chen L, Yang J (2022) Sound-based construction activity monitoring with deep learning. Buildings. https://doi.org/10.3390/buildings12111947

Yang Y, Peng Z, Zhang W, Meng G (2019) Parameterised time-frequency analysis methods and their engineering applications: a review of recent advances. Mech Syst Signal Process 119:182–221. https://doi.org/10.1016/j.ymssp.2018.07.039

Yang F, Jiang Y, Xu Y (2022a) Design of bird sound recognition model based on lightweight. IEEE Access 10:85189–85198. https://doi.org/10.1109/ACCESS.2022.3198104

Yang Z, Wang Y, Pan Y, Huan R, Liang R (2022b) Inaudible sounds from appliances as anchors: a new signal of opportunity for indoor localization. IEEE Sens J 22:23267–23276. https://doi.org/10.1109/JSEN.2022.3211098

Yu X, Li X (2023) Sound recognition method of coal mine gas and coal dust explosion based on GoogLeNet. Entropy 25:412. https://doi.org/10.3390/e25030412

Zaman K, Sah M, Direkoglu C, Unoki M (2023) A survey of audio classification using deep learning. IEEE Access 11:106620–106649. https://doi.org/10.1109/ACCESS.2023.3318015

Zawawi SA, Hamzah AA, Majlis BY, Mohd-Yasin F (2020) A review of MEMS capacitive microphones. Micromachines 11:484. https://doi.org/10.3390/mi11050484

Zelios A, Grammenos A, Papatsimouli M, Asimopoulos N, Fragulis G (2022) Recursive neural networks: recent results and applications. SHS Web Conf 139:03007. https://doi.org/10.1051/shsconf/202213903007

Zhang Z, Zhang R, Li Z, Bengio Y, Paull L (2020) Perceptual generative autoencoders. In: Proceedings of the 37th international conference on machine learning. PMLR, pp 11298–11306

Zhang Z, Shen Y, Valdes JJ, Huq S, Wallace B, Green J, Xi P, Goubran R (2023) Domestic sound classification with deep learning. In: 2023 IEEE sensors applications symposium (SAS). pp 01–06. https://doi.org/10.1109/SAS58821.2023.10254050

Zhang Z, Liu H, Shao Y, Yang J, Liu S, Yuan G (2024) CFENet: a contrastive frequency-sensitive learning method for gas-insulated switch-gear fault detection under varying operating conditions using acoustic signals. Eng Appl Artif Intell 135:108835. https://doi.org/10.1016/j.engappai.2024.108835

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

Authors and Affiliations

Pau Gairí[1] · Tomàs Palleja[1] · Marcel Tresanchez[1]

✉ Pau Gairí
pau.gairi@udl.cat

Tomàs Pallejà
tomas.palleja@udl.cat

Marcel Tresanchez
marcel.tresanchez@udl.cat

[1] Research Group in Logic, Optimization and Robotics, Department of Industrial Engineering and Building, Universitat de Lleida, Jaume II, 69, 25001 Lleida, Spain