Internet of Things 26 (2024) 101153

Contents lists available at ScienceDirect

Internet of Things

journal homepage: www.elsevier.com/locate/iot

Review article

Internet of Intelligent Things: A convergence of embedded systems, edge computing and machine learning

Franklin Oliveira [a], Daniel G. Costa [b,*], Flávio Assis [a], Ivanovitch Silva [c]

[a] Graduate Program in Mechatronics, Federal University of Bahia, Salvador, Brazil
[b] SYSTEC-ARISE, Faculty of Engineering, University of Porto, Porto, Portugal
[c] Department of Computer Engineering and Automation, Federal University of Rio Grande do Norte, Natal, Brazil

ARTICLE INFO

Keywords:
TinyML
IoT
Hardware
Artificial intelligence
Survey

ABSTRACT

This article comprehensively reviews the emerging concept of Internet of Intelligent Things (IoIT), adopting an integrated perspective centred on the areas of embedded systems, edge computing, and machine learning. With rapid developments in these areas, new solutions are emerging to address previously unsolved problems, demanding novel research and development paradigms. In this sense, this article aims to fulfil some important research gaps, laying down the foundations for cutting-edge research works following an ever-increasing trend based on embedded devices powered by compressed artificial intelligence models. For that, this article first traces the evolution of embedded devices and wireless communication technologies in the last decades, leading to the emergence of IoT applications in various domains. The evolution of machine learning and its applications, along with associated challenges and architectures, is also discussed. In this context, the concept of embedded machine learning (TinyML) is introduced within the context of the Internet of Intelligent Things paradigm, highlighting its unique characteristics and the process of developing and deploying such solutions. Furthermore, we perform an extensive state-of-the-art survey to identify very recent works that have implemented TinyML models on different off-the-shelf embedded devices, analysing the development of practical solutions and discussing recent research trends and future perspectives. By providing a comprehensive literature review across all layers of the Internet of Intelligent Things paradigm, addressing potential applications and proposing a new taxonomy to guide new development efforts, this article aims to offer a holistic perspective on this challenging and rapidly evolving research field.

1. Introduction

In the 1980s, mainframes were the primary means of information processing, although they had limited computational power and were not easily accessible when compared to current standards. As technology progressed, these devices were miniaturised and gave way to personal computers (PCs), which introduced the concept of individualised computing. While office applications were the most commonly used on PCs, they also had the capability to control the physical environment of workplaces. These innovations inspired Mark Weiser to coin the term *Ubiquitous Computing*, envisioning computers that could process information anytime and anywhere [1]. Weiser also foresaw the integration of computers into everyday products, resulting in their pervasive and "invisible" presence today, from embedded systems in cars and home appliances to a myriad of gadgets available in our daily lives [2].

* Corresponding author.
E-mail addresses: franklinoliveira@ufba.br (F. Oliveira), danielgcosta@fe.up.pt (D.G. Costa), fassis@ufba.br (F. Assis), ivanovitch.silva@ufrn.br (I. Silva).

https://doi.org/10.1016/j.iot.2024.101153
Received 4 November 2023; Received in revised form 29 February 2024; Accepted 1 March 2024
Available online 4 March 2024

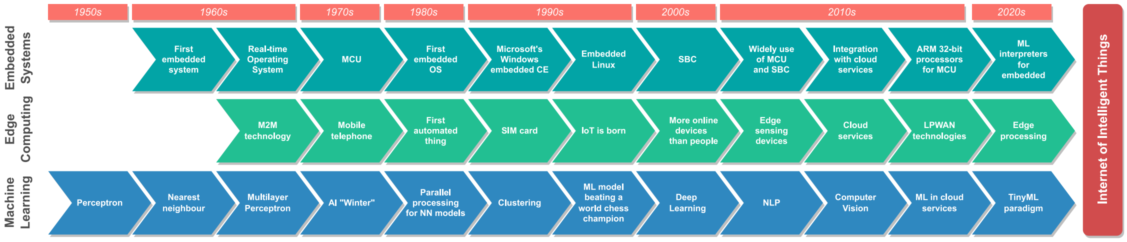

Fig. 1. The convergence of related areas to the development of the Internet of Intelligent Things.

Weiser's prediction could be applied in the real world thanks to the emergence of embedded devices such as microcontrollers, which have become "ubiquitous" in the sense they could transparently interact with other devices through wireless technologies [3]. These communication technologies, such as LoRaWAN, Bluetooth, Wi-Fi, and 5G, are increasingly common nowadays, being a key factor in the emergence of the concept of the Internet of Things (IoT). In short, the so-called IoT paradigm seeks to integrate the processing power of information and communication into everyday objects that were previously not equipped with computational technology [4].

In a broader perspective, the IoT paradigm has evolved over three conceptual generations, with the first one being characterised by machine-to-machine communications using resources like tagged objects and RFID identifiers, and then progressing to a second generation based on communications toward cloud-based services [5]. More recently, the third generation introduced the widely-used Three-Layer Architecture, consisting of a perception layer where devices collect sensing data on-site, a network layer that transfers information from devices to an application layer, and the application layer itself, which is typically located in the cloud and responsible for processing the data received from the network layer [4,6]. This conceptual architecture has created the foundations for more widespread developments of IoT-based services that we have seen in the last decade.

When realising the three aforementioned generations, two major concepts have been distinguished: cloud and edge. The former is a generic concept that relies on increased computing power and high storage capacity to provide enhanced-centred services to remote users, using communication protocols for request/response procedures. In contrast, the latter depends on devices capable of perceiving the environment and processing associated data locally, which might then be transmitted to cloud-based computing servers for storage or post-processing actions [7]. Although particular architectures may employ the cloud and edge concepts differently, there is a well-consolidated perception that edge-based elements are closer to the physical world and the end users, while cloud-based elements are implemented in multiple remote servers that behave like a unified entity [8]. Since there are some cost, energy, and time constraints in applications that are dependent on cloud servers [9], the Internet of Intelligent Things (IoIT) paradigm has been promoted as a way to perform more elaborated processing tasks in edge devices, potentially enhancing performance as a whole. Actually, this could enable previously passive agents in the system to become more active, redistributing the processing burden of modern IoT systems and affecting networking bandwidth and timeliness requirements [10,11]. This way, the Internet of Intelligent Things comes as promising convergence of embedded hardware, wireless networking, and artificial intelligence elements, setting the foundations for a series of disruptive applications in areas such as Smart Cities, Industry IoT, Smart Homes, Agriculture 4.0, Smart Healthcare, among others [12–17].

In the evolution of IoT applications, which has more recently favoured edge-based approaches, the existence of affordable hardware development platforms has been crucial. Regarding the current hardware landscape, microcontroller-based embedded devices now boast more computational power, thanks to their use of 32-bit processors with enhanced RAM and Flash memories, while single-board computers (SBCs) incorporate 64-bit processors and may even include dedicated low-end GPUs, all usually following an energy-efficient design and with affordable retail prices [2]. Consequently, the emerging IoIT paradigm has been able to be increasingly reliant on new processing paradigms that support efficient edge computing, with Artificial Intelligence (AI) algorithms receiving significant attention. As edge devices with adequate hardware resources can now extract information from tabular, visual, and audio data accurately and reliably, the execution of Machine Learning AI algorithms on small, affordable devices has become an integral part of the design of intelligent IoT systems [18].

When putting all these elements together, the TinyML paradigm emerges as a game-changer when considering the constrained resources of edge devices for executing artificial intelligence algorithms. By embedding compressed AI models capable of performing ML functions like linear regression and classification, TinyML can revolutionise the processing of sensed data in IoT-based applications. The evolution of edge computing, driven by advancements in embedded devices that leverage low-power wireless network technologies, has recently empowered limited-processing-power devices to effectively employ ML models for solving diverse problems. Furthermore, the field of Machine Learning now offers reliable architectures optimised for TinyML, further solidifying this trend. In fact, recent years have already witnessed the transformative potential of embedded ML principles in revolutionising various applications and achieving remarkable outcomes through cost-effective solutions, as will be discussed in this article.

In Fig. 1, we draw an evolution timeline for some of the most relevant embedded systems, edge computing, and machine learning landmarks, depicting important conceptual phases that have led to the highly anticipated Internet of Intelligent Things revolution.

Roughly speaking, IoIT emerges when IoT devices meet embedded AI algorithms. However, while it holds promise for numerous benefits, various challenges have also surfaced. Therefore, gaining insight into the current state-of-the-art and anticipated development trends in this field becomes crucial. This article then provides a comprehensive survey of the Internet of Intelligent Things paradigm, offering a holistic perspective. By doing so, it aims to expand upon previous review works, such as [18–20], which focused on this research field following a "limited" review approach: while the works in [18,19] are centred on the execution of machine learning models on microcontrollers, the work in [20] brings contributions on TinyML development within traditional IoT scenarios. In general, those previous surveys cover important aspects when bringing AI closer to the edge, but our article goes further by addressing IoIT across all of its layers, presenting existing and forthcoming challenges, as well as some perspectives envisioned for this field. Finally, a more up-to-date perspective of the literature is presented.

The remainder of this article is organised as follows. Section 2 presents the background of the Internet of Things, approached from the perspective of the evolution of embedded devices and wireless communication technologies that have enabled the development of IoT applications that are already present in our daily lives. In Section 3, the evolution of the Machine Learning area is presented, as well as its current challenges and architectures. In Section 4, the concept of the Internet of Intelligent Things is formally presented from the relationship with TinyML, showing the particularities of this area, as well as the definition of Intelligent IoT device classes, and its optimised implantation criteria. Section 5 deals with the presentation of recent research works developed in this area, along with future perspectives on the development of applications based on Intelligent IoT. Then, final discussions are presented in Section 6, followed by conclusions and references.

2. The rise of the Internet of Things

In large cities around the world, modern technology is taking centre stage, bringing exciting changes to the way we live. In this scenario, embedded systems are becoming more and more prevalent, collaborating and offering a variety of cutting-edge applications and services for residents, businesses, and governments [21]. When properly leveraged, embedded technologies may support the development of the Internet of Things (IoT) paradigm, which envisions promoting the ubiquitous presence of a variety of devices capable of monitoring and/or acting in order to achieve a shared purpose, usually in a collaborative way [6].

Generally speaking, most IoT applications perform their tasks following the definitions of a three-layer architecture. The first layer is composed of embedded systems, which deal with low-power, dependable, real-time equipment for monitoring the sensed environment. The second layer is communication, which involves sharing information with the third (application) layer and between devices, usually through wireless connections, with the latter providing the actual services for any requester. To make this logical operation more practical, wireless networks and embedded computer technology had to advance over many years, facing a lot of challenges on multiple scales. The development of those fields is discussed in this section.

2.1. Embedded systems evolution

The journey toward the current state of IoT applications, which now finds relevance in various scenarios such as urban, domestic, and medical settings, has been a gradual evolution with many milestones in roughly five decades of advancements in embedded systems. Fig. 2 depicts this evolution process since the launch of humanity's first embedded device, the Apollo Guidance Computer (AGC), in the 1960s, culminating in the development of affordable hardware development boards.

The AGC was the first computer to use silicon integrated circuits (ICs) and it is regarded as the first embedded device, guiding, navigating, and controlling the spacecraft that transported Neil Armstrong and Buzz Aldrin in the Apollo 11 mission in 1969. Considered an engineering marvel, this computer weighed around 30 kg and had a processor with a frequency of 2 MHz that interpreted assembly code [22]. After this engineering milestone, efforts turned to the miniaturisation of electronic components that resulted in the emergence of the first microprocessors between the 60s and 70s, until the creation of the first microcontrollers, such as the Intel 8048. Launched in 1976, that is the most prominent MCU in the MCS-48 family, and compared to the AGC, it has a 3 MHz higher processor frequency, in addition to being about 20x smaller and lighter [23].

Following the development of the first MCUs as a result of component downsizing, the primary goal became to enhance the resources of these embedded devices. During this time, microcontrollers began to be available with 8, 16, and 32-bit processors, as well as frequencies of up to 32 MHz. Furthermore, analogue-to-digital converters, serial ports, timers, and interruptions were incorporated into these devices, resulting in a significant boost in the capacity to read sensors and process data in real-time for a variety of applications, mainly in the context of Wireless Sensor Networks (WSN) [24].

WSNs gained popularity in the mid-2000s as a first step in the development of IoT applications, with the introduction of the MicaZ (2003) and TelosB (2006) boards being notorious [25]. These embedded devices feature an IEEE 802.15.4 wireless communication interface, making them the first boards to support remote monitoring for a wide range of applications. It should be noted, however, that these devices were more expensive than those currently available and had a higher learning curve for programming, with even proprietary programming languages, as is the case with TelosB [26].

Due to the restricted flexibility of the boards used in WSNs, the developed applications were somehow limited in sensing capability and data processing, mostly relying on remote servers when effectively using the sensed data. This scenario, on the other hand, prepared the way for a new development era based on more accessible and easier-to-program embedded devices (using programming languages like C and C++), resulting in the first models of boards widely recognised and utilised in the present community, such as Arduino and STM32. These boards provided better affordability and programming flexibility, accelerating the development of IoT applications [23].

However, because these embedded devices lacked a network interface, remote contact with applications or other devices was mostly impossible. Single Board Computers (SBCs) appeared then as embedded devices more powerful than microcontrollers, with

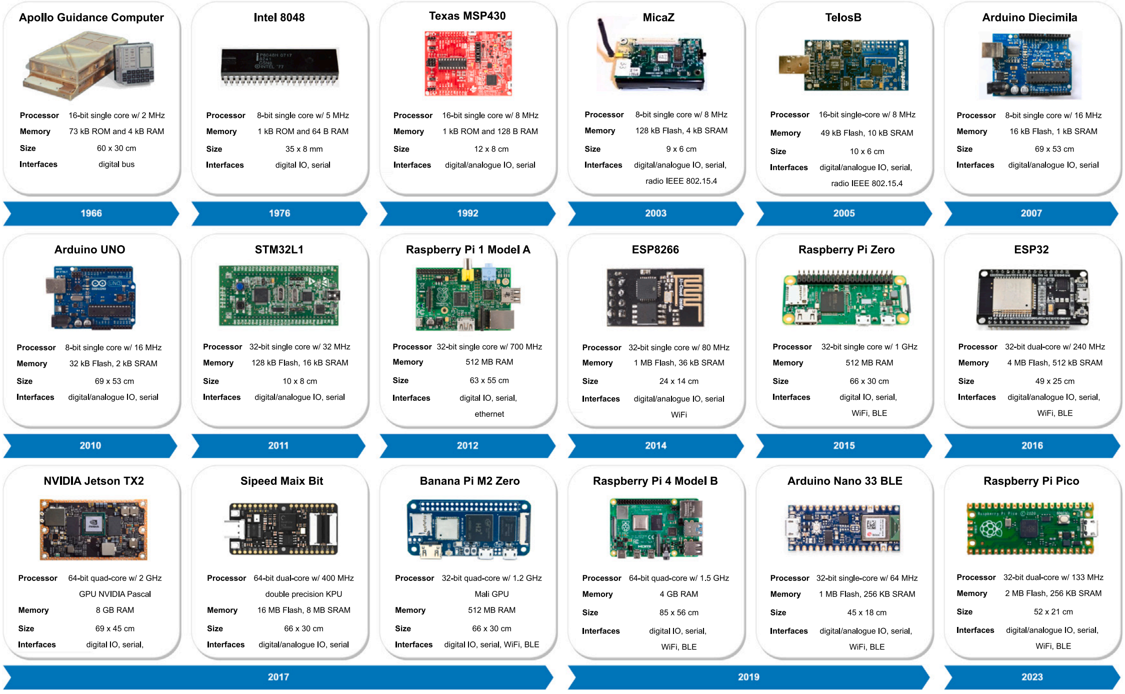
Apollo Guidance Computer
Processor 16-bit single core w/ 2 MHz
Memory 73 kB ROM and 4 kB RAM
Size 60 x 30 cm
Interfaces digital bus
1966
Intel 8048
Processor 8-bit single core w/ 5 MHz
Memory 1 kB ROM and 64 B RAM
Size 35 x 8 mm
Interfaces digital IO, serial
1976
Texas MSP430
Processor 16-bit single core w/ 8 MHz
Memory 1 kB ROM and 128 B RAM
Size 12 x 8 cm
Interfaces digital/analogue IO, serial
1992
MicaZ
Processor 8-bit single core w/ 8 MHz
Memory 128 kB Flash, 4 kB SRAM
Size 9 x 6 cm
Interfaces digital/analogue IO, serial, radio IEEE 802.15.4
2003
TelosB
Processor 16-bit single-core w/ 8 MHz
Memory 49 kB Flash, 10 kB SRAM
Size 10 x 6 cm
Interfaces digital/analogue IO, serial, radio IEEE 802.15.4
2005
Arduino Diecimila
Processor 8-bit single core w/ 16 MHz
Memory 16 kB Flash, 1 kB SRAM
Size 69 x 53 cm
Interfaces digital/analogue IO, serial
2007
Arduino UNO
Processor 8-bit single core w/ 16 MHz
Memory 32 kB Flash, 2 kB SRAM
Size 69 x 53 cm
Interfaces digital/analogue IO, serial
2010
STM32L1
Processor 32-bit single core w/ 32 MHz
Memory 128 kB Flash, 16 kB SRAM
Size 10 x 8 cm
Interfaces digital/analogue IO, serial
2011
Raspberry Pi 1 Model A
Processor 32-bit single core w/ 700 MHz
Memory 512 MB RAM
Size 63 x 55 cm
Interfaces digital IO, serial, ethernet
2012
ESP8266
Processor 32-bit single core w/ 80 MHz
Memory 1 MB Flash, 36 kB SRAM
Size 24 x 14 cm
Interfaces digital/analogue IO, serial WiFi
2014
Raspberry Pi Zero
Processor 32-bit single core w/ 1 GHz
Memory 512 MB RAM
Size 66 x 30 cm
Interfaces digital IO, serial, WiFi, BLE
2015
ESP32
Processor 32-bit dual-core w/ 240 MHz
Memory 4 MB Flash, 512 kB SRAM
Size 49 x 25 cm
Interfaces digital/analogue IO, serial, WiFi, BLE
2016
NVIDIA Jetson TX2
Processor 64-bit quad-core w/ 2 GHz GPU NVIDIA Pascal
Memory 8 GB RAM
Size 69 x 45 cm
Interfaces digital IO, serial,
Sipeed Maix Bit
Processor 64-bit dual-core w/ 400 MHz double precision KPU
Memory 16 MB Flash, 8 MB SRAM
Size 66 x 30 cm
Interfaces digital/analogue IO, serial
Banana Pi M2 Zero
Processor 32-bit quad-core w/ 1.2 GHz Mali GPU
Memory 512 MB RAM
Size 66 x 30 cm
Interfaces digital IO, serial, WiFi, BLE
2017
Raspberry Pi 4 Model B
Processor 64-bit quad-core w/ 1.5 GHz
Memory 4 GB RAM
Size 85 x 56 cm
Interfaces digital IO, serial, WiFi, BLE
Arduino Nano 33 BLE
Processor 32-bit single-core w/ 64 MHz
Memory 1 MB Flash, 256 KB SRAM
Size 45 x 18 cm
Interfaces digital/analogue IO, serial, WiFi, BLE
2019
Raspberry Pi Pico
Processor 32-bit dual-core w/ 133 MHz
Memory 2 MB Flash, 256 KB SRAM
Size 52 x 21 cm
Interfaces digital/analogue IO, serial, WiFi, BLE
2023

Fig. 2. The evolution of hardware technologies and development boards in five decades.

single-core processors running at hundreds of MHz, memories starting at 256 MB, and the ability to run Linux-based operating systems. Furthermore, these SBCs typically included USB, HDMI, and jack ports, allowing the use of peripherals such as a mouse, keyboard, and monitor on these tiny boards [2].

Since Ethernet requires a wired connection, IoT applications based on hardware boards with an RJ45 connector (such as the first generation of Raspberry Pi boards) were still limited in terms of device placement and outdoor deployments. Around 2014, the MCU ESP8266 was released in a trend to change this configuration, which, in addition to having more processing power and memory than previous MCUs, also featured a (IEEE 802.11) WiFi interface, allowing embedded applications to connect to the Internet. After this, in 2015, the Raspberry Pi Zero, which also has a WiFi interface and Bluetooth Low Energy (BLE) capability, was released bringing an additional perspective to the development of IoT applications [23].

At this time, the advancement of embedded devices since the AGC was already immense, with boards much smaller and much more powerful reaching processing frequency in GHz for the SBCs and tens of MHz for the MCUs. However, there was still opportunity for progress in such a short time, and this was capitalised on with the 2016 release of the ESP32, the first MCU with a dual-core CPU at a frequency of hundreds of MHz [23]. Soon after, in 2017, some of the first boards with a graphics processing unit were released, including the quad-core SBCs NVIDIA Jetson TX2 (64-bit) and Banana Pi M2 Zero, which are equipped with NVIDIA and ARM GPUs. In terms of MCUs, we could also consider the Sipeed Maix Bit board, which features a 64-bit dual-core processor and a double precision KPU (Knowledge Processor Unit), making it suitable for efficiently running machine learning algorithms via its graphics processing units [27].

Finally, in 2019, the evolution of embedded devices started to focus on improving their peripherals, with the release of the Raspberry Pi 4, a SBC model with a 1.5 GHz quad-core processor and available with 4, 6, and 8 GB memory, and the Arduino Nano 33 BLE and Raspberry Pi Pico MCUs, both equipped with 32-bit processors and WiFi and BLE connectivity. More recently, the Raspberry Pi 5 came offering new features and enhanced processing power, reinforcing the trend for powerful edge processing at affordable prices. As a result, the progress of embedded devices to the present day has resulted in the release of many hardware development boards that are still available today, even in different versions, having processing capability to enable lightweight machine learning architectures and wireless network connectivity, enabling a variety of Intelligent IoT applications [4]. This scenario has been exploited in different ways, opening many possibilities when enabling embedded machine learning-based applications on the edge.

2.2. The advent of wireless networks

Over five decades, embedded devices have undergone significant evolution in processing power, memory resources, energy efficiency, size, programming flexibility, and cost. In parallel, wireless networks, one of the main pillars of IoT, have also experienced

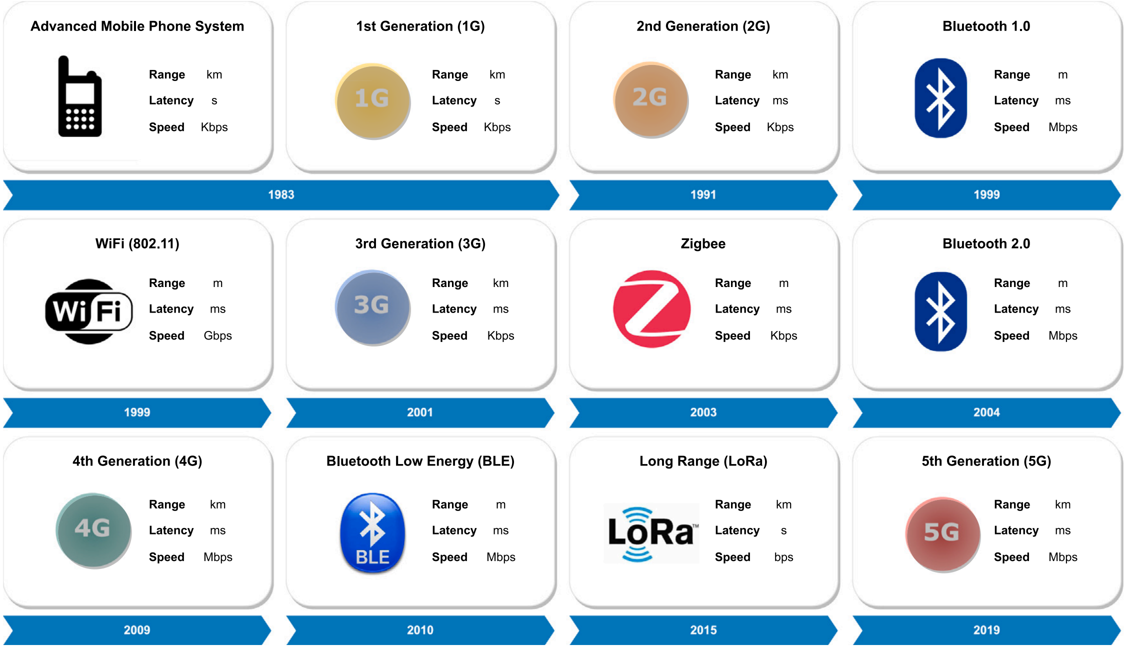

Fig. 3. The evolution of wireless networks.

progress on their own, roughly starting from 1983 with the Advanced Mobile Phone System (AMPS), the first communication technology for analogue mobile telephony. The AMPS played a crucial role in making the idea of cellular telephony commercially viable, allowing users to connect across a wide geographic area [28]. When the first generation (1G) of cellular networks appeared, as depicted in Fig. 3, it enabled voice communication over analogue modulation to send signals between cell towers and mobile devices, a great breakthrough in that time. This was a major milestone, allowing people to communicate while on the move [29].

Wireless networking for mobile telephony continued advancing with the development of new technologies. The introduction of 2G (1991) enabled mobile devices to deliver text messages, while Bluetooth 1.0 (1999) enabled file transfers between devices over short distances. IEEE 802.11 WiFi (1999) offered high-speed Internet connections over short distances, and 3G (2001) enabled mobile Internet connections, as well as voice calls and messaging, expanding on previous generations' features [30].

In 2003, an innovative wireless network technology aimed at communicating with embedded devices was introduced: the Zigbee. Based on the IEEE 802.15.4 standard, this technology establishes efficient and low-energy communication, which enables innovative home and industrial automation applications. Examples of Zigbee-enabled embedded devices included the previously mentioned MicaZ and TelosB boards (Fig. 2), released in 2003 and 2005, respectively. These boards exemplify Zigbee's ability to provide reliable and efficient communication for a variety of embedded applications [31]. For the embedded system world, more efficient and affordable wireless technologies meant greater opportunities to expand the application scenarios.

In general, wireless networking technologies have been leveraged to allow communications among embedded devices, enabling the creation of WSN and IoT applications. For some development boards, extra hardware modules might be inserted to support different wireless network technologies including Bluetooth 2.0 (2004) and 4G (2009) [32]. However, as described in Section 2.1, embedded devices with built-in WiFi connectivity started to appear in 2014, accelerating the creation of boards with similar features, like the ESP8266. With this, the proliferation of IoT applications was facilitated, which allowed embedded devices to communicate with other devices and servers [32], deeply transforming the way ubiquitous IoT applications would be developed.

Several wireless network technologies with special characteristics that could benefit embedded devices have arisen, taking into account some requirements such as low-energy consumption, ad hoc data transmission, and long-range communications. These technologies include LoRaWAN, which was announced in 2015, and Bluetooth Low Energy (BLE), which was introduced in 2010. Currently, LoRaWAN is frequently utilised in IoT applications to connect devices across long distances. However, it is important to point out that LoRaWAN has some limitations in terms of latency and speed, which can affect real-time communication of large amounts of data [33].

Overall, some well-known limitations for IoT devices are being (partially) overcome with the arrival of 5G in 2019, the fifth generation of high-speed, low-latency mobile communication networks. The fifth generation of mobile networks was designed to primarily enable real-time IoT applications, offering an efficient and up-to-date solution not only for the context of the Internet of Things but also wireless communications in general. Finally, faster and more dependable communication is anticipated with 5G, allowing for the instantaneous integration of devices and the introduction of cutting-edge services in a variety of industries, notably for applications that use Machine Learning algorithms in embedded devices for data processing in cloud services, which is still widely used [34].

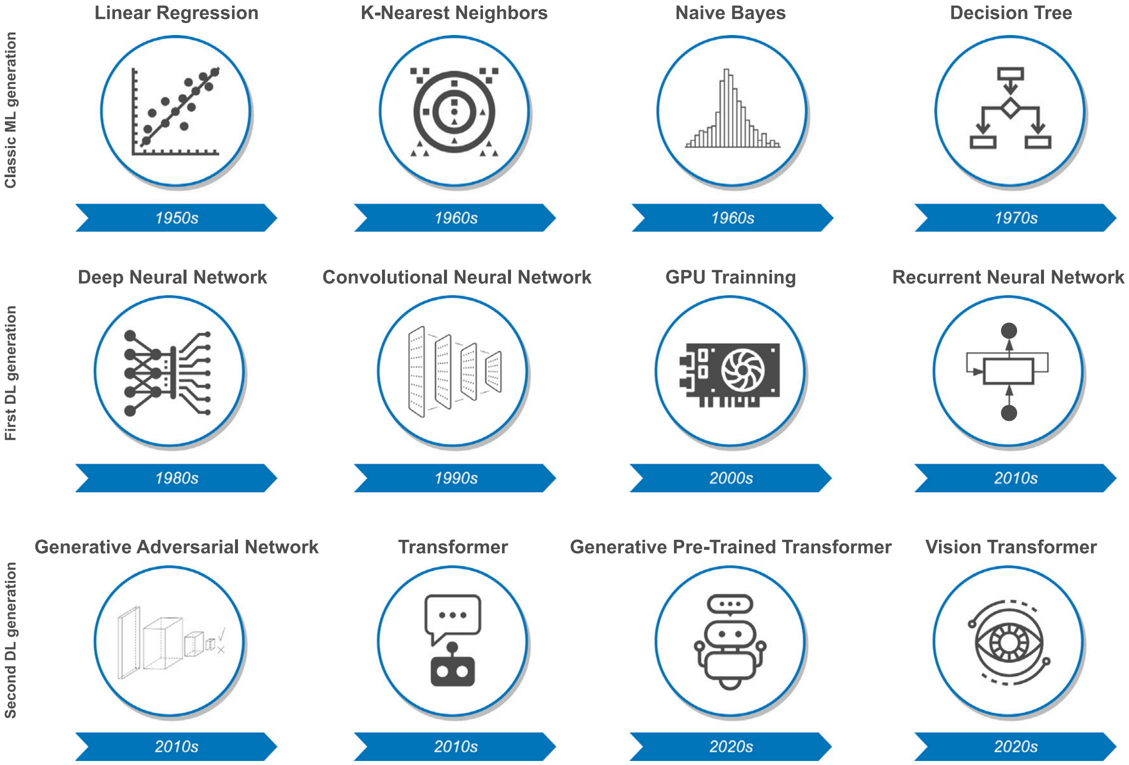

Fig. 4. The evolution of the Machine Learning field over five decades.

As a last remark, while IoT devices are increasingly adopting new wireless communication technologies as cost reduces and hardware embedding is facilitated, energy consumption, high bandwidth and reduced latency demands are still some limitations that should restrain how fast new game-changing technologies like 5G will be adopted in practical applications. This is even more apparent when considering the anticipated advancements of 6G communications, which will eventually succeed 5G. At first glance, the 6G technology aims to provide even faster data speeds, lower latency, greater capacity, and more reliable connections compared to its predecessor [35,36]. Overall, this will be a deeper revolution for IoIT, particularly when considering that 6G is being designed to enhance Massive Machine Type Communications (mMTC), while also supporting innovative AI-driven networking. While 5G adoption is not yet widespread, its benefits for applications like Smart Cities and Industry 4.0 are widely acknowledged. However, the prospects for 6G are even more promising, especially considering the escalating demands for bandwidth and latency in an increasingly machine-to-machine communication-dependent landscape. The maturation of IoIT stands to benefit significantly from these advancements, paving the way for unprecedented levels of connectivity and efficiency across various industries and domains. Nevertheless, since 6G should not be around in this decade, 5G is already expected to unleash a series of profound transformations for the still-evolving Internet of Intelligent Things.

3. The machine learning revolution

The Machine Learning (ML) "concept" has undergone a significant evolution since its beginnings, with initial developments in the so-called "classic ML" until the impressive achievements in the Deep Learning era. In this classical ML, algorithms were mainly based on statistical models and supervised and unsupervised learning techniques [37]. In short, supervised learning is a conceptual approach in which algorithms are trained with input data and their respective desired outputs, aiming to learn a model that can generalise and make correct predictions (*inferences*) for new unlabelled data [38]. On the other hand, unsupervised learning involves analysing unlabelled data, seeking to identify patterns, structures, or clusters within the provided data set, without the need for prior information about classes or target values [39]. Both approaches have been widely exploited in classical ML to solve problems of classification, clustering, regression, and feature extraction, allowing a more interpretive understanding and analysis of data [37,40].

Fig. 4 presents the temporal evolution of different Machine Learning algorithms and paradigms, highlighting an evolution trend that has unravelled in parallel to the previously discussed areas of embedded systems and wireless networking.

As depicted in Fig. 4, some of the main classical ML algorithms include Linear Regression, Logistic Regression, Decision Trees, K-Nearest Neighbours, Support Vector Machines, and Naive Bayes, which have been applied in a variety of domains such as fraud detection, demand forecasting, product recommendation, sentiment analysis, and pattern recognition [37], just to cite some of the most common applications. Actually, this vast set of possible applications for classical ML algorithms directly benefits from their inherent simplicity and easiness of explanation (also referred to as “explainability”), making it easier to grasp the results and draw insights from the data [41]. However, limitations in their ability to handle complicated and non-linear data also pose some restrictions, which prompted the development of more advanced systems like Deep Learning [42].

As a culminating stage of the mentioned evolution trend, Deep Learning (DL) has emerged as a new paradigm based on neural networks capable of automatically learning complicated data representations at many levels of abstraction. These networks are usually composed of numerous layers of processing units called artificial neurons that conduct mathematical operations on input data to change it into more meaningful representations. Each succeeding layer pulls more abstract elements from the data, enabling the detection of subtle patterns and correlations [42]. Obviously, more elaborated modelling of a particular system or a set of variables comes at the cost of higher computational costs, which is one of the reasons why Deep Learning has become more popular when more affordable and powerful hardware resources come into the scene [43].

The capacity of DL to automatically learn important features rather than relying on manual feature engineering, as is the case with traditional ML techniques, is one of its key advantages [44]. This means that deep neural networks can handle unprocessed, high-dimensional data, including text, audio, and image data, and instantly learn representations that are more valuable for some application domains. Deep neural networks are also very adaptable and can be trained on massive datasets, using the processing capacity of a GPU (Graphics Processing Unit) to accelerate training, particularly creating more generic models after the 2000s (Fig. 4) [45]. These characteristics have made deep learning especially effective in areas such as Natural Language Processing (NLP), using Recurrent Neural Networks, computer vision, applying Convolutional Neural Networks, and many other high-end applications [42,46]. By learning complex and hierarchical representations of data, deep neural networks have achieved remarkable performances in tasks such as machine translation, object recognition, sentiment detection, and medical diagnosis, leading to significant advances in several fields [42].

Currently, the field of machine learning is still developing thanks to cutting-edge deep learning models like Generative Adversarial Networks (GANs). These GANs have revolutionised content creation by making it possible to produce realistic images and movies using training data. This important development landmark has strengthened graphic design and computer vision, opening the door for original and inventive applications in both domains [47]. Simultaneously, the emergence of the attention mechanism and the use of transformers have played crucial roles in machine learning nowadays [48]. These approaches have been particularly successful in natural language processing applications, allowing the development of advanced models of Large Language Models (LLMs) such as the GPT (Generative Pre-trained Transformer). Notable instances, such as ChatGPT, which is based on the GPT architecture, present an astonishing ability to interpret and synthesise coherent text, which is accelerating the creation of smarter and more conversational virtual assistants [49].

In parallel, machine learning has made great progress in computer vision, leading to the development of techniques like the Vision Transformer (ViT). With the help of attention and transformations, ViT has achieved success in image analysis, enabling the extraction of pertinent characteristics [50]. Segment Anything (SAM) is a technique that employs ViT to precisely and effectively segment items into images, advancing object recognition and analysis in the field of computer vision, and it is a noteworthy example of this [51]. These recent advancements in the field of machine learning are propelling the development of more advanced systems capable of understanding, generating, and analysing information in increasingly sophisticated ways, with practical applications in a wide range of fields such as natural language processing, computer vision, and many others. This constant evolution is opening up new opportunities and leading the machine learning field forward, which is currently being employed in a variety of areas such as health, finance, industry, etc [52,53].

However, until the early 2020s, the need for substantial computing power for modern machine learning was closely tied to cloud-based infrastructure. This approach has provided scalable and flexible capabilities, enabling companies and organisations to run ML algorithms efficiently, handle large volumes of data, train complex models, and run inferences in real time. Cloud computing also promotes collaboration and sharing of models and data across teams and organisations, driving the advancement and adoption of machine learning in many areas of research and application, without requiring significant investments in on-premises infrastructure [54]. Nevertheless, the associated costs of these complex infrastructures, notably the related carbon footprint in a world increasingly concerned with sustainable solutions, have also raised important concerns about the widespread adoption of complex ML models strongly dependable on cloud-based computational power. Actually, with the promotion of edge AI solutions and TinyML algorithms, it is expected that a large set of problems can be addressed by compressed tailored solutions with reduced energy consumption, potentially opening a more sustainable development trend for ML [55,56].

Conversely, the recent deployment of ML-based approaches at the edge has enabled the birth of a new generation of IoT, the Internet of Intelligent Things [4]. For this, model optimisation techniques have been created and enhanced to enable the compression and reduction of models with the goal of implanting them in microcontrollers and single-board computers, making them “intelligent”. These strategies began to emerge with the introduction of TinyML in the early 2020s, as well as the development of embedded devices, as discussed in Section 2.1. However, IoIT also faces some challenges, such as available computing resources, energy supply, and storage limitations on devices, which may continue to limit the implementation of many ML architectures on embedded devices. Moreover, there is a need to manage and update locally deployed models, necessitating the definition of a path to idealise the efficient deployment of models in embedded devices, optimising their resources, and actually paving the way for the rising of the Internet of Intelligent Things paradigm [18].

4. Foundation of the Internet of Intelligent Things

The growing demand for smarter devices, able to understand contexts and make decisions on the spot instead of relying exclusively on systems in the cloud, has driven the development of the next generation of IoT-based applications [10,11]. This new generation, referred to as the Internet of Intelligent Things, is still experiencing its first steps, but the expectations are already high. In a nutshell, this ongoing revolution aims to create solutions that operate with greater autonomy and efficiency, with interconnected embedded devices being able to perform complex tasks and quickly adapt to the users' needs [4]. Moreover, when surpassing the conventional reasoning of machine-to-machine communications previously enabled by the Internet of Things paradigm, the IoIT innovates by bringing artificial intelligence closer to the IoT devices, defining a new operation scope. As a result, we can expect that the IoIT will be the leading paradigm for new disruptive applications in many scenarios, moving the Internet of Things paradigm to a more prominent place.

Generally speaking, the aforementioned development trend toward IoIT has been supported by the convergence of various computing areas, described as follows:

- The **edge computing** paradigm, which benefits from current embedded devices with enough computational power to process large volumes of data;
- The "classical" **internet of things**, which establishes the infrastructure for interconnecting devices that will work together to achieve common goals;
- The emerging **TinyML** concept, which allows the implementation of machine learning models (both classical ML and Deep Learning) in embedded devices such as microcontrollers and single board computers, giving them the perception of "intelligence" [57].

As expected, the main challenges faced by the new generation of IoIT applications are directly related to the execution of artificial intelligence algorithms in embedded devices, which will typically have limited computational resources. In this context, TinyML emerges as a key element, allowing the implementation of efficient and compact machine learning models, suitable to run on these constrained devices. In this way, overcoming these challenges is crucial to further drive the development of the Internet of Intelligent Things, enabling it to offer intelligent and adaptable solutions that benefit various areas of everyday life [18].

Given the high complexity of machine learning models, which generally necessitate devices with high computational power and, in some cases, a dedicated GPU, the main impediments to the evolution of the Internet of Things from the paradigm of "monitoring - transmission" to "monitoring - comprehension - transmission" are related to the accurate and energy-efficient deployment of Machine Learning (ML)/Deep Learning (DL) models in embedded devices [4].

The primary obstacle stems from the nature of classical IoT applications, which rely on the deployment of numerous networked devices to solve common problems. In this setting, these devices must be affordable, which inevitably leads to devices with minimal computational power and memory resources. This limitation of computing resources makes implementing ML/DL models on IoT devices more difficult than the currently popular paradigm, where models reside in the cloud, typically employing robust computers with dedicated GPUs [2].

Therefore, it is critical to define a cutoff point for selecting the embedded devices that will form the basis for Internet of Intelligent Things applications. We define three guiding criteria for that:

1. **C1**: Energy efficiency, allowing for sustainable, prolonged, flexible, and potentially secure operation;
2. **C2**: Affordability, seeking for large-scale deployment;
3. **C3**: Sufficient processing capacity, supporting the execution of ML/DL models.

All of these factors are critical and must be taken into account in Internet of Intelligent Things projects. For example, in some cases, it may be critical to avoid blanket adoption of NVIDIA Jetson boards, because while they meet criterion C3, they may fall short of requirements C1 and C2. Similarly, designing IoIT devices based on Arduino UNO can be difficult because it may match criteria C1 and C2, but fall short of criterion C3. As a result, prudent device selection based on these three criteria is critical in this context.

At this point, a reference taxonomy is proposed to relate classes of embedded devices to the implementation of ML/DL models, using the concept of TinyML as the main guiding point. This taxonomy classifies devices into two categories: "High-end TinyML", consisting of embedded devices with greater computational power, but not excessive to the point of disregarding criteria C1 and C2, and "Low-end TinyML", which encompasses embedded devices with lower computational power, but sufficient for a growing group of small tasks that require some level of "intelligence".

4.1. High-end TinyML

This group encompasses all embedded devices that have reasonably high computational power within the IoIT scope. In general, they will have the ability to execute the most robust TinyML models while still respecting deployment criteria C1 and C2. As a result, these devices are highly valued according to criterion C3, while maintaining remarkable energy efficiency and a moderate cost that makes them viable for large-scale deployment.

Based on the aforementioned criteria, the class of embedded devices that fits into the High-end TinyML category is known as Single-board Computers (SBCs). The SBCs are entire, working computers that are integrated on a single printed circuit board and contain all of the main components of a computer, such as a processor, RAM, storage, and multiple input and output interfaces, such as USB, HDMI, Ethernet, and Wi-Fi, among others. Furthermore, it is typical for SBCs to support entire operating systems, such as Linux, Windows, or Android, which broadens their capabilities and possibilities for use in TinyML projects. Due to their versatility, compact size and energy efficiency, SBCs have become a prominent choice for applications that require more computing power in resource-constrained environments [58]. Their ability to run robust TinyML models in real-time and their ease of integration across multiple devices make them ideal for implementing advanced artificial intelligence solutions in our defined context.

SBCs serve an important role as sensor nodes in the IoT landscape, gathering data from diverse sources such as temperature, humidity, brightness, and acceleration [59]. These sensor nodes are usually dispersed throughout several surroundings, allowing for distributed monitoring and data collection. Furthermore, SBCs are commonly employed as IoT gateways, serving as data intermediates between sensor nodes and central processing and storage systems. This gateway capability enables effective data aggregation and transmission to the cloud or other processing infrastructures. In this sense, SBCs are especially indicated for the implementation of models that require greater computational power, making them ideal for more complex tasks, such as computer vision (object detection and tracking, semantic segmentation, pose estimation) and speech processing (speech recognition, natural language processing).

Therefore, these embedded devices may be used to solve problems in critical IoIT scenarios [60,61]: their ability to collect data from sensors in a distributed manner and transmit it efficiently, coupled with their ability to process complex artificial intelligence models, makes SBCs a versatile and powerful choice to meet the challenges of these diverse environments. In smart cities, for example, they help with traffic monitoring, energy resource management, and smart urban services. In precision agriculture, SBCs are key to optimising the use of agricultural resources and improving production and sustainability. In industry 4.0, these devices enable intelligent automation, machine monitoring and process optimisation, contributing to greater efficiency and productivity [58,62]. In summary, the use of SBCs in intricate IoIT scenarios greatly accelerates digital transformation and makes it possible for smarter, more connected, and effective solutions across a range of industries. The Raspberry Pi Zero, 2, 3, 4, and 5 boards, the Banana Pi M2 Zero, as well as the NVIDIA Jetson TX2 board, which represents a high-performance option albeit at a higher cost (moving away from criterion C2), are some of the best-known SBCs that fall into the High-end TinyML class that were relevant examples presented in Section 2.1.

In general, it is reasonable to say that the possibilities for deploying cutting-edge solutions in many IoIT scenarios have increased as a result of the availability of SBC boards, revolutionising the development of this area [62].

4.2. Low-end TinyML

While the High-end TinyML class is more associated with the criterion C3 when implementing applications in the Internet of Intelligent Things context, we define that the Low-end TinyML class is more associated with criteria C1 and C2, ensuring better energy efficiency and lower implementation cost, as opposed to lower computational performance (being weaker in criterion C3). This class is composed of extremely low-power embedded devices, known as Microcontrollers (MCUs).

MCUs differ from SBCs in terms of their CPU architectures, which has a big impact on how powerful they are computationally. With low power and energy-efficient processors, they have a small number of resources, including low RAM, integrated flash storage, and simple I/O interfaces, and are highly integrated and designed for particular embedded applications [2]. In fact, the computational power difference between SBCs and MCUs is obvious, putting the Low-end TinyML class in a position where it can run simpler ML/DL models, making it ideal for the implementation of simple artificial intelligence solutions. As a result, this type of device has been extensively used for simpler monitoring activities, mostly employing structured data from analogue sensors [63].

There are many examples of successful IoIT applications when employing MCUs. In health monitoring, MCUs may be used for the detection of irregularities or alarms in the event of a medical emergency via biometrics sensor data, which are often employed in wearable devices. In environmental monitoring, MCUs may be used for collecting and processing data from sensors that indicate pollution levels and air quality. Additionally, Low-end TinyML devices also favour the concept of the Internet of Intelligent Things when pursuing the Industry 4.0 concept, easing the implementation of solutions to monitor and control machines while assuring higher flexibility and scalability [64]. This can usually be performed by the continuous monitoring of multiple variables, trying to identify failures and allowing preventive maintenance when the sensed data is processed using ML/DL models. Another example more associated with logistics is based on the adoption of MCUs to classify products and packages through simpler computer vision architectures, facilitating the process of separating products in a factory [4].

Microcontrollers can be adopted with more modern CPU architectures and greater processing capacity. The ESP32, Arduino Nano 33 BLE, and the Raspberry Pi Pico are popular choices that provide the necessary processing power for the implementation of Low-end TinyML models. Additionally, some options could be viewed as outliers among MCUs, with a high computational power due to the use of a specific processing unit for ML/DL as FPUs (Floating-Point Unit), well fulfilling criterion C3 of implantation of TinyML models, but distancing themselves from criteria C1 and C2 due to the tendency of these MCUs to be pricey and energy-intensive (as is the case with Sipeed Maix Bit and Arduino Portenta H7).

As a result, MCUs have moved from a position where they were used exclusively as sensor nodes in past IoT generations into a position where it is possible to process information and make decisions based on TinyML models, becoming intelligent sensor nodes [65,66]. This trend can be perceived when comparing some devices of the Low-end TinyML class with the devices of the High-end TinyML class, since some of them overlap when concerning our defined comparison criteria, even partially. In order to highlight these facts, Table 1 summarises the most popular boards for both classes in our taxonomy.

Table 1
Low-end and High-end TinyML devices' specifications.

	Board	Cost (US$)	Supply (A)	RAM (MB)	Arch.	Cores	Freq. (MHz)	Add.
Low-end	Arduino Nano BLE 33	~25	~0.5	0.256	32-bit	1	64	
	Adafruit EdgeBadge	~36	~1.0	0.192	32-bit	1	120	
	ESP32 DevKitC	~10	~0.5	0.512	32-bit	2	240	
	Raspberry Pi Pico W	~7	~0.5	0.256	32-bit	2	133	
	STM32F746	~15	~0.5	0.5	32-bit	1	216	
	Sipeed Maix Bit	~45	~1.0	8.0	64-bit	2	600	FPU
	SparkFun Edge	~17	~0.5	0.384	32-bit	1	96	FPU
High-end	Banana Pi M2 Zero	~20	~2.0	512	32-bit	4	1200	GPU
	Orange Pi Zero 3	~35	~2.0	1000	64-bit	4	1500	GPU
	Raspberry Pi Zero W	~15	~1.2	512	32-bit	1	1000	
	Raspberry Pi Zero 2 W	~20	~1.2	512	64-bit	4	1000	
	Raspberry Pi 3 model B	~35	~2.5	1000	64-bit	4	1200	
	Raspberry Pi 4 model B	~35	~3.0	1000	64-bit	4	1800	
	Raspberry Pi 5	~56	~5.0	4000	64-bit	4	2400	
	NVIDIA Jetson Nano	~99	~5.0	4000	64-bit	4	1430	GPU

Finally, in relation to High-end TinyML class devices, it is possible to observe in Table 1 that they tend to have approximately 10x higher energy consumption than low-end devices, reaching up to 5 A. In contrast, computational power also tends to be approximately 10x higher, with CPUs with up to 4 cores at 64-bit and 2.4 GHz. Another point that can be observed is the availability of memory in devices of this class, which can be thousands of times greater than those of low-end devices, with RAM of 512 MB and 1 GB, but with versions that support up to 2, 4, and even 8 GB. In terms of cost, the Zero line boards (Banana and Raspberry) fit criterion C2 well, costing between $15 and $20, which is close to the cost of some of the surveyed microcontrollers, being attractive possibilities for a genuine implementation with more processing capability. On the other hand, the Jetson Nano, NVIDIA's entry-level card, is impracticable for some applications due to its extremely high cost and energy consumption, meeting only criterion C3 very well, as previously discussed.

4.3. State-of-the-art applications and current research scenario

In order to answer this question in an initial effort, a survey of the literature was conducted about the development of TinyML applications between the period of 2020 and 2023, highlighting the novelty of this concept. In this investigation, we considered some of the most relevant research databases as reference, namely IEEE Xplore, Springer Link, ScienceDirect, ACM Digital Library, and MDPI, utilising the following search query: "Machine Learning" & "Embedded" & "Edge" & "Internet of Things".

During this search, we identified a total of 58 papers published in relevant conferences and journals that met our keywords. On that, we performed a stringent screening process, prioritising papers that either mentioned or effectively integrated ML/DL models on boards classified as Low-end or High-end, encompassing not only the model training and conversion processes. Subsequent to this rigorous screening process, we curated a final selection of 27 research papers, which are summarised in Table 2. In that Table, "Deploy" indicates whether the respective research work implemented some deployment strategy and employed ML models on embedded devices, while "Benchmark" indicates whether the corresponding work conducted evaluations pertaining to resource consumption on embedded devices, encompassing considerations such as memory, CPU, or energy utilisation.

After performing this survey, some conclusions could be taken. First, the following characteristics could be identified:

- The majority of the works used supervised learning models like Convolutional Neural Networks (CNNs) and Deep Neural Networks (DNNs), but some unsupervised learning models like Isolation Forest were also identified;
- Applications are more prevalent in the medical, industrial, and agricultural contexts, employing sensors such as cameras, microphones, and scalar sensors for image classification, detection, and segmentation, as well as sound detection, classification, and regression;
- Among the Low-end class boards presented in Table 1, the most identified in the performed survey was ESP32, STM32, and Arduino Nano 33. The Raspberry Pi Pico board has been little explored, being a gap for investigation since it is the cheaper MCU board and with lower estimated energy consumption;
- Among the High-end TinyML boards, there is a greater use of Raspberry board models 3 and 4, which are the ones with the highest costs and energy consumption. The use of the Zero line boards is also little explored;
- The model's accuracy, memory usage, and latency were essentially evaluated every time embedded AI was adopted. Few studies examined energy usage, and none examined the board's resources and consumption over an extended time while simulating a real application.

In addition to the listed observations, in Table 2 it is also possible to observe that the most used framework for the implementation of TinyML models is TensorFlow in its two variations: TensorFlow Lite (TFLite), for high-end devices, and TensorFlow Lite for Microcontrollers (TFLM), for low-end devices. However, in recent years, a new trend has been to use TinyMLOps platforms, such as Edge Impulse (EI), which performs training, optimisation, and prepares the code for deployment on the target device [90], making

Table 2
Literature on TinyML-based applications between 2020 and 2023.

Work	Year	Field	Class	Deploy	Benchmark	Board	Sensors	Problem	Framework	Learning	Architecture
[67]	2020	–	Low-end	x	x	STM32, ESP32, Adafruit Feather, Adafruit Metro	Camera	Image classification	TFLM	Supervised	SVM
[68]	2020	Robotics	High-end	x	x	Jetson AGX	Camera	Image segmentation	TensorRT	Supervised	CNN
[14]	2021	Medical	Low-end			–	Microphone	Sound detection	TFLM	Supervised	ANN
[69]	2021	Industrial	High-end	x		Jetson Nano	Camera	Anomaly detection	–	Supervised	GAN
[70]	2021	Industrial	Low-end	x		Arduino Nano 33	Vibration	Online learning	TFLM	Unsupervised	DNN
[71]	2021	Agriculture	High-end	x		Raspberry Pi 3	Camera	Object detection	PyTorch	Supervised	CNN
[72]	2021	Vehicular	Low-end	x		Arduino Nano 33	IMU	Anomaly detection	–	Unsupervised	TEDA
[73]	2021	–	Low-endHigh-end	x	x	STM32, ESP32, Adafruit Feather, Adafruit Metro, etc	Camera	Image classification	TFLM	Supervised	SVM
[74]	2021	Traffic	Low-end			Arduino UNO	Piezoeletric	Regression	SciKit	Supervised	Random Forest
[75]	2022	Smart city	Low-end High-end	x	x	ESP32, Raspberry Pi 4, Google Coral	Camera	Object detection	TFLMTFLite	Supervised	CNN
[76]	2022	Vehicular	High-end	x		Jetson Nano	Camera	Object detection	PyTorch	Supervised	CNN
[77]	2022	Vehicular	High-end	x		Jetson TX2	Microphone	Sound detection	–	Supervised	DNN
[78]	2022	Industrial	Low-endHigh-end	x	x	Arduino Nano 33 Raspberry Pi 4	Temperature	Classification	TFLMTFLite	Supervised	MLP
[79]	2022	Robotics	Low-end	x	x	Raspberry Pi Pico	Camera	Image segmentation	TFLM	Supervised	CNN
[80]	2022	Medical	Low-end	x		Arduino Nano 33	Microphone	Spectogram classification	EI	Supervised	CNN
[16]	2022	Energy	High-end	x		Raspberry Pi 4	Camera	Image classification	TFLite	Supervised	CNN
[15]	2023	Industrial	Low-end	x	x	ESP32	IMUTemperature	Anomaly detection	–	Unsupervised	Isolation Forest
[81]	2023	–	Low-end	x	x	STM32	Camera	Image classification	–	Supervised	CNN
[82]	2023	Medical	Low-end	x		Arduino Nano 33	ECG	Classification	TFLM	Supervised	CNN
[83]	2023	Vehicular	Low-end			–	Camera	Image classification	TFLM	Supervised	CNN
[84]	2023	Agriculture	Low-end	x		Arduino Nano 33	Spectral	Classification	EI	Supervised	CNN
[85]	2023	Industrial	High-end	x		Raspberry Pi Zero	WiFi RSSI	Classification	EI	Supervised	DNN
[86]	2023	Home	Low-end	x		Arduino Nano 33	Gas	Classification	EI	Supervised	DNN
[87]	2023	Robotics	High-end	x		Jetson Nano	Robotics	Image segmentation	–	Supervised	DNN
[88]	2023	Medical	Low-end	x		Arduino Nano 33	Camera	Image classification	EI	Supervised	CNN
[12]	2023	Automotive	Low-end			–	Fuel	Regression	–	Unsupervised	TEDA
[89]	2023	Energy	Low-end	x		Arduino Nano 33	Camera	Image classification	EI	Supervised	CNN

the process a “black box”. However, while these works engage in substantial discussions about the AI models they adopt, a critical research gap emerges in the lack of proper evaluation of the hardware constraints of the employed boards.

Therefore, it is clear that the question posed in this section cannot be fully addressed. This is due to the hardware utilisation in scenarios where AI is embedded, which may need to operate uninterruptedly over an extended duration. Elaborating on the significance of addressing this question, it is imperative to underscore that the Internet of Intelligent Things represents a transformative paradigm in the realm of interconnected devices. The practical feasibility of IoIT applications heavily relies on the ability to balance the demands of intelligent decision-making with the constraints of hardware resources, especially in prolonged operation scenarios. This way, resolving this question is pivotal for the advancement of IoIT, as it informs the design and optimisation of systems that aim to strike a harmonious equilibrium between intelligent functionality and resource sustainability, ultimately shaping the future of connected, intelligent ecosystems. We believe that this question is still to be better answered in the coming years.

5. Applications and perspectives of IoIT

The construction of the Internet of Intelligent Things paradigm revolves around the integration of intelligent devices equipped with embedded artificial intelligence and wireless networking capabilities, as discussed in previous sections. These devices are designed to perform a wide range of tasks and function autonomously, making them a pivotal component of the evolving digital landscape. In recent years, the literature has witnessed a surge in research works highlighting various applications of IoIT, each accompanied by its own unique set of challenges and contributions. This section will shed light on some of these noteworthy applications, underlining their significance and the innovations they bring to the forefront of technology and connectivity.

5.1. IoIT application domains

The Internet of Intelligent Things paradigm has rapidly expanded its footprint across a multitude of application domains, showcasing its transformative potential in various sectors. In recent years, some new application areas have emerged, depicted as follows:

- Smart Cities: IoIT plays a pivotal role in the development of smart cities by creating interconnected ecosystems that have the potential to enhance urban living in multiple ways. Intelligent devices, such as smart traffic lights, environmental sensors, and autonomous transportation, may work to optimise different urban systems, particularly when performing decisions on the edge [55,91]. With stakeholders investing more resources to better prepare for the negative impacts of climate change, it is natural to expect that IoIT may come as an affordable tool to accelerate the desired smart city transformation into more sustainable and resilient urban areas;

- Industry 4.0: IoIT has been expected as the driving force behind Industry 4.0 [92,93]. Intelligent machines, equipped with AI and IoT sensors, enable real-time monitoring and predictive maintenance, increasing production efficiency, minimising downtime, and reducing operational costs. In the coming years, this trend should be reinforced;
- Smart Agriculture: in general, agriculture has seen a remarkable transformation through the adoption of IoIT. Smart farming solutions leverage AI-powered sensors and drones to monitor crops, soil conditions, and livestock health [94]. When supported by data-driven approaches, IoIT may enhance crop yields, conserve resources, and ensure sustainable agricultural practices, even when brought to an urban agriculture scenario [95];
- Military Applications: the military sector may harness IoIT to gain a tactical advantage through enhanced situational awareness. Intelligent drones, wearable devices, and smart logistics systems will bolster the capabilities of armed forces, improving communication, surveillance, and strategic decision-making [96]. When embedded AI is also considered, the possibilities to better understand the field and react properly are considerably enhanced, bringing tactical advances for military uses;
- Space Exploration: space agencies have been employing IoIT to advance space exploration missions, especially due to cost, reliability, and efficiency issues [97]. Intelligent spacecraft and rovers have been equipped with AI-driven systems that autonomously navigate and perform complex tasks on distant celestial bodies, expanding our understanding of the target environments;
- Healthcare: IoIT has revolutionised healthcare with smart medical devices and remote patient monitoring. Wearable health trackers and AI-powered diagnostics have been used to enable early disease detection and personalised treatment plans, improving patient outcomes and reducing healthcare costs [98]. This trend might still be very prominent, with embedded AI evolving to provide more accurate and fast solutions for many healthcare issues;
- Environmental Monitoring: the adoption of IoIT to gather data that can be associated with the tackling of climate change is an important trend, with intelligent sensors playing an important role. In parallel, the monitoring of pollution levels and or urban variables has also been considered. In general, intelligent sensors and satellite networks will keep helping scientists to make informed decisions to mitigate environmental threats and protect ecosystems, reinforcing the relevance of IoIT in this scenario [99];
- Energy Management: IoIT can be leveraged to optimise energy consumption in both residential and industrial settings. Smart grids, connected appliances, and energy-efficient buildings enable users to reduce energy waste, lower costs, and reduce their carbon footprint [100];
- Education: IoIT has the potential to revolutionise education with smart classrooms and personalised learning experiences. Intelligent devices and AI-driven educational tools may address the individual student needs, enhancing the quality of education in multiple ways [101].

In each of these domains, IoIT presents unique challenges and contributions. Recent works have exploited low-end and high-end hardware boards with embedded AI algorithms, with solutions evolving as hardware capabilities improve and the efficiency of the AI algorithms increases. In general, the fusion of intelligent devices with AI promises a future marked by greater efficiency, sustainability, and quality of life across a multitude of sectors.

5.2. TinyMLOps and the IoIT revolution

The endeavour of designing smart devices that encompass the requisites of IoIT, covering the previously discussed application domains, presents intrinsic challenges. In general, it is critical to examine a variety of devices deployed in various localities, where all of these devices require frequent and effective updating of their ML/DL models in order to maintain intelligence levels with accurate results. This operational environment necessitates the creation of a Machine Learning Operations flow (Machine Learning Operations - MLOps) tailored for distributed and embedded environments, also known as TinyMLOps (Tiny Machine Learning Operations) [102].

When considering a typical workflow for developing and provisioning TinyML models on IoIT devices, it is feasible to break it down into two steps closely associated with the locations where the tasks are performed, namely, in the cloud layer and on IoIT devices. This way, Fig. 5 presents a generic TinyMLOps pipeline highlighting the expected services for both conceptual layers, which can be further adjusted by different applications according to their particular requirements.

The phase of the TinyMLOps pipeline performed in the cloud encompasses operations aimed at developing the model, following the same process of the MLOps. This phase begins with the data Extraction, Transformation, and Loading (ETL) procedure, which can be presented in structured, image or audio formats. This ETL results in the composition of a dataset that encompasses both the training set and the test set, which are used in later stages [103]. Subsequently, the training set from this dataset is used to fine-tune a ML/DL model with a specific architecture during the Training stage. This culminates in the creation of a model precisely adapted to the IoIT application context, which is evaluated in the Validation stage to identify whether the performance of its responses is in line with what is expected from a set of test data [104].

This initial phase of the pipeline, which consists of training and validating the ML/DL model, represents the stages of the TinyMLOps process that presents the lowest performance risk when it comes to deployment on devices belonging to the Internet of Intelligent Things. This is because the model will not be adapted to the embedded context, limiting this phase to the learning process,

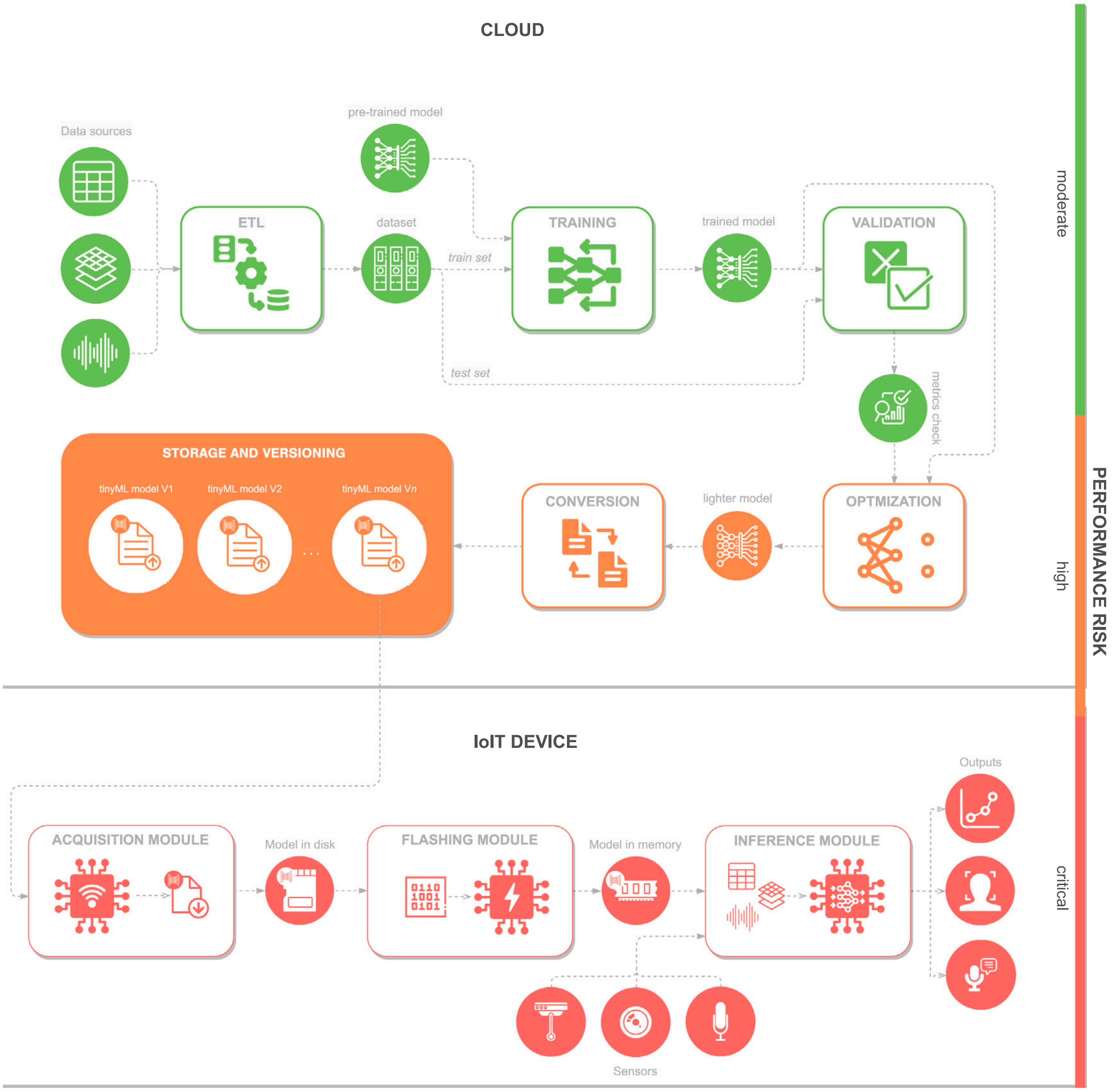

Fig. 5. Representation of the TinyMLOps cycle to create IoIT devices.

particularly in supervised learning settings [105]. Once the model has been customised to the specifics of the IoIT application context, the process of converting it to the TinyML domain continues, with the goal of distributing it to the IoIT devices that comprise the application's perception layer.

The process of adapting the model to the TinyML context is primarily an optimisation process in which the model is subjected to modifications in order to improve its efficiency and make it agile and lightweight. These optimisations are accomplished through the use of techniques such as quantisation and pruning [106], and they represent a critical stage due to their ability to determine the model's performance when running on IoIT devices, particularly in terms of computational resource consumption [107]. In this regard, this step is distinguished by the presence of significant risks, both in terms of TinyML model performance and the potential deterioration of knowledge acquired in previous phases, because inadequate optimisations have the potential to damage the model's accuracy [108]. However, if the model is successfully optimised, resulting in a lighter ML/DL model, the next step is to convert it to the TinyML context, followed by its insertion in the cloud storage and versioning module, as presented in Fig. 5.

In order to better highlight these ideas, Table 3 presents some of the frameworks commonly adopted in a typical TinyMLOps pipeline.

As summarised in Table 2, some works have employed the Edge Impulse framework to implement TinyML solutions [90]. Actually, although not specified in Table 3, the Edge Impulse already implements all steps of the TinyMLOps pipeline, gaining popularity lately.

Table 3
Some popular TinyMLOps frameworks.

	Stage	High-end	Low-end
Cloud	Training	TensorFlow, PyTorch	TensorFlow, Edge Impulse
	Validation	TensorFlow, PyTorch	TensorFlow, Edge Impulse
	Optimisation	Pruning, Quantisation (optional)	Pruning, Quantisation (necessary)
	Conversion	TensorFlow Lite, ONNX	TensorFlow Lite for Microcontrollers, Edge Impulse
IoIT device	Flashing module	TensorFlow Lite Runtime, ONNX Runtime	Over-the-air
	Inference module	TensorFlow Lite Runtime, ONNX Runtime	TensorFlow for Microcontrollers Library

Once the TinyML models are available in the cloud, the TinyMLOps cycle proceeds on to the second and final phase, which is the delivery of the models to IoIT devices for actual use on the edge. In order to provide a reference implementation in this scenario, as presented in Fig. 5, these devices should implement three generic modules that include both hardware and embedded software:

1. **Acquisition Module**: consists of hardware with wireless network interfaces that allow direct communication with the TinyML model storage and versioning module in the cloud. It also includes the embedded software required for authentication in this service as well as downloading the representative file of the most recent model, which is then saved on disk;
2. **Flashing Module**: responsible for transferring the TinyML model previously downloaded from the cloud to the local memory of the IoIT device, preparing it for use. Generally, this module consists of libraries specialised in loading models, as well as supporting inference execution;
3. **Inference Module**: includes specific libraries for loading TinyML models into memory and moving the model from memory to the processing unit (CPU or GPU, if available). Furthermore, it maintains sensor inputs capable of creating data in the model's specific context, which may include scalars, images, or audio. Finally, it generates the model response, allowing IoT devices to intelligently process input.

Given the full TinyMLOps cycle, with an emphasis on developing and delivering models to enable IoIT devices, it is important to highlight that this generic process incorporates specific technologies and techniques aimed at the two categories of devices outlined in this article, namely, the low-end and high-end. These differences apply only to the Training, Validation, Optimisation, and Conversion stages in the cloud phase, while in the IoIT device phase, they are concentrated in Flashing and Inference modules, as presented in Table 3.

In conclusion, the integration of TinyMLOps into the context of IoIT devices is a multifaceted journey, intricately weaving together specialised technologies like TensorFlow Lite [109], ONNX runtime [110], OTA (Over-the-air) [111], and more. As delineated in this section, this process is not an one-size-fits-all solution; rather, it adapts to the unique demands of low-end and high-end devices at various stages, from the cloud-based training and optimisation to the IoIT device-specific flashing and inference. By harnessing these diverse tools and techniques, organisations can navigate the ever-evolving landscape of IoIT with finesse, delivering efficient and intelligent devices that cater to a wide spectrum of applications and use cases, followed by some specific challenges which current researches aim to solve.

6. Discussion

After conducting an in-depth review of the literature and subsequent analyses, several research and development guidelines emerge. These guidelines serve as valuable indications of persistent challenges and promising research trends, supporting future works. In this section, we summarise the challenges of the presented convergence of embedded systems, edge computing, and machine learning, revisiting key aspects to be considered. The main strategies and promising technical solutions to address these challenges are also identified. Finally, perspectives and research directions are envisioned, paving the way for new developments in this area.

6.1. Research challenges

The previously discussed convergence of embedded systems, edge computing, and machine learning within the realm of the IoIT presents a multitude of challenges, each with its own set of constraints and expected obstacles. The primary identified challenges can be summarised as follows:

- Resource constraints: there is a need for proper reasoning of the computational resource constraints of the employed electronics. Edge devices will typically operate with limited computational power, memory, and energy resources, making it challenging to execute complex machine learning algorithms properly. In this sense, the balancing of computational resources, energy efficiency and affordability will be a common guiding factor in IoIT applications;
- Adaptability: this is related to the possibility of IoIT deployment on multiple different locations, changing the context of the monitored scenarios. Nowadays, context switching presents itself as an important challenge to be tackled in most applications. IoT scenarios are usually defined by the presence of countless interconnected devices distributed across diverse geographic locations and operating in a wide range of contexts, which necessitate a comprehensive approach to the adaptation and

customisation of ML/DL models. The inherent complexity of these ecosystems is further complicated by their dynamic and changing nature, where situations can change quickly. As a result, edge models require adaptability in order to satisfy the demands of ever-changing IoT scenarios [112];
- Security and data preservation: ensuring data privacy and security in IoIT deployments is crucial but challenging. Processing sensitive data at the edge necessitates robust security measures to safeguard against potential breaches or unauthorised access. Nevertheless, the additional complexity of ensuring security may conflict with existing resource constraints and adaptability requirements;
- Interoperability: this is another pressing challenge with increasing relevance in new applications. It was discussed that IoIT ecosystems comprise heterogeneous devices and platforms, requiring seamless communication and interoperability to facilitate efficient data exchange and collaboration. In this scenario, scalability is also a concern, as IoIT systems must accommodate a growing number of connected devices and handle increasing data volumes effectively. Furthermore, some applications demand real-time processing at the edge, imposing strict latency constraints and further complicating resource management and optimisation efforts.

The existence of these challenges highlights the necessity to keep improving IoIT systems. Overall, these challenges may become obstacles to the deployment of intelligent devices in this new era of IoT, but some perspectives are promising to alleviate them.

IoIT involves deploying machine learning models directly on edge devices for local data processing, thereby reducing latency and bandwidth usage. Adopted computing approaches will distribute computing tasks across edge devices, optimising resource utilisation and scalability. Moreover, federated learning may enable collaborative model training across distributed edge devices while preserving data privacy, which is a critical consideration in IoIT environments.

In this article, we defined a taxonomy to categorise the adopted embedded devices: "High-end" and "Low-end". The criteria for classification are energy efficiency, affordability, and sufficient processing capacity, which we identified as fundamental characteristics. As discussed before, while high-end devices have greater computational power, low-end devices have lower computational power but are sufficient for small tasks that require some level of intelligence. In this sense, an important consideration is the use of hardware acceleration, such as the use of a GPU (Graphics Processing Unit), TPU (Tensor Processing Unit), and FPGA (Field Programmable Gate Array), which will typically enhance the performance of machine learning inference on edge devices. Software optimisation techniques, including model compression, quantisation, and pruning, may also help to reduce the computational and memory requirements of machine learning models, making them suitable for deployment on resource-constrained edge devices. All these aspects are crucial to achieving the expected goals within the defined budget.

It is also noticeable that each enabling technology for IoIT offers distinct advantages and limitations. Edge computing, for instance, reduces latency and improves privacy by processing data locally but is constrained by limited computational power and storage capacity compared to cloud servers. Machine learning at the edge enables real-time decision-making and preserves data privacy but faces challenges in optimising models for edge deployment. Wireless communication technologies facilitate seamless connectivity between edge devices but are susceptible to bandwidth limitations and security vulnerabilities. Future works should then be concentrated in reducing the gap between the expected performance and the existing hardware/software landscape, taking advantage of new technological tools and methods.

Since we want to enable the implementation of efficient and compact machine learning models suitable for running on resource-constrained devices, TinyML will be a guiding point. However, hardware and software elements must be considered in a combined perspective, since the development and deployment of compressed artificial intelligence models is also critical for the limited computational resources of embedded devices. These models should be designed to be lightweight and have reduced memory and processing requirements while maintaining acceptable accuracy levels.

6.2. Perspectives and future directions

The ongoing advancements in the Internet of Intelligent Things offer valuable insights into promising research trends. In fact, these insights serve as crucial guideposts for the evolution of this application domain. This subsection compiles various perspectives and research directions gathered from the performed literature review and discussions.

From an initial perspective, knowledge transfer emerges as a machine learning research challenge, as models developed in a particular context may not be easily relevant in another. In order to adjust to contextual differences, the ability to employ incremental learning principles becomes critical. In this context, Federated Learning comes as a promising approach that has attracted increasing attention in this regard. In fact, it enables models to be trained locally on edge devices using their data, reducing the need for centralised data aggregation that might be impractical or even impossible in some IoT scenarios [113].

Adopting federated learning within the Internet of Intelligent Things paradigm offers several compelling benefits. Firstly, it addresses privacy concerns by enabling model training directly on edge devices without the need to transmit sensitive data to centralised servers, thus preserving data privacy and security. Secondly, federated learning reduces the communication burden on networks by leveraging local computation, making it particularly suitable for IoT environments with limited bandwidth or intermittent connectivity. This approach also enhances scalability as it distributes the computational load across numerous edge devices, allowing for the training of large-scale models without overwhelming individual devices or network infrastructure. Doing so, federated learning may promote adaptability by enabling models to be fine-tuned on diverse data sources, reflecting the unique characteristics and contexts of individual devices or user preferences. Nevertheless, while there are advantages, it also poses challenges that include increased complexity in training and aggregation, particularly due to heterogeneity among edge devices. For the IoIT landscape, however, the compelling opportunities may still favour the adoption of federated learning in many scenarios, notably in efficient 5G and 6G networking settings [114,115].

Approaches like federated learning may help to mitigate the challenges of concept drift, but they may require high computational and energy costs, which contrasts with lower-cost and long-range technologies, such as LoRaWAN [113]. In this way, the investigation of concept drift in different locations of IoIT devices is a highly relevant academic challenge in the machine learning area, being recently addressed in the literature [116–118].

The federated learning concept also needs to deal with the limited computational resources of IoIT devices, which may be ill-suited for both model training and inference tasks due to their computational resource constraints [119]. Typically, for this type of "learning", there is a scenario in which the inference devices belong to a class of devices with limited resources, while the training devices constitute another class with more capacity intended exclusively for training. Although the federated learning approach allows inference devices to collaborate when training models locally, the problem of scaling training from multiple inference devices to just one more powerful training device becomes evident [120]. This is due to the need for maintaining training efficiency since synchronising and aggregating models from distributed inference devices into a single training device can overwhelm it, which makes scalability an even more complex issue [121]. At this point, variations like Split Learning have emerged to enable collaborative model training without sharing raw data between devices. Split learning is particularly well-suited for scenarios where data privacy is a concern, which can be a valuable addition to the toolbox of machine learning techniques for IoIT applications [115].

In addition to the lack of computational resources in embedded devices, little is known about the capacity of conventional platforms to execute TinyML models for long periods. Moreover, it may be not straightforward to know which architectures and tasks can be supported and how they behave in a real scenario. This aspect makes device deployment in the Internet of Intelligent Things problematic, being an inherently embedded systems challenge [122]. Due to the fundamental nature of these devices, processing capacity and available memory are severely limited. Because of these limitations, conventional machine learning and deep learning architectures are frequently beyond the reach of these embedded applications, opening up space for comprehensive research in the field of optimising existing architectures, as well as creating lightweight and efficient architectures aimed specifically at implementation in the context of IoIT [123].

Nevertheless, although it is unquestionable that these resource-constrained devices lack the necessary capacity to perform inferences based on high computational cost architectures, there is a notable lack of comprehensive benchmarking studies that thoroughly address a variety of popular embedded hardware platforms, in addition to evaluating a diverse range of ML/DL architectures applied to specific tasks in areas such as computer vision, audio processing, data reduction, among others [122]. The lack of this type of study constitutes a significant gap in current knowledge, emphasising the urgent need for thorough studies that provide a clear and comparative view of the capabilities of various embedded devices in terms of ML/DL implementation. Doing so, it would be feasible to identify the optimal scenarios for employing ML/DL technologies in IoIT devices, as well as to guide the creation of optimised and personalised architectures to satisfy the specific demands of these constantly evolving devices [124].

In summary, the integration of intelligence into IoT devices opens up a plethora of applications across sectors like medical, agriculture, automotive, and smart cities. However, challenges such as limited computational resources, context switching, security, and the need for adaptation in dynamic environments, persist. Federated learning may be promising in addressing some of these challenges but it also introduces new obstacles related to computational constraints and scalability. Furthermore, benchmarking studies are lacking in evaluating embedded hardware platforms for ML/DL implementations, highlighting the need for comprehensive research in this area. Ultimately, overcoming these challenges is crucial for the successful deployment of intelligent IoT devices, ensuring that they meet quality criteria such as energy efficiency, affordability, and adequate processing capacity for real-world applications across various domains.

7. Conclusions

The Internet of Intelligent Things is a deliberate and ever-changing convergence of embedded systems, edge computing, and machine learning. In this field, the integration of these three elements is critical for developing increasingly intelligent and efficient solutions that allow linked devices and systems to interact, make autonomous decisions, and provide novel services. This convergence reflects an evolution in the aforementioned domains that began in the 1950s, encompassing a still ongoing quest to fully use the potential of interconnection between embedded devices, data processing near to the points of origin, and improved machine learning algorithms.

In the current scenario, we have embedded platforms with sufficient computational resources to guarantee processing at the edge via Machine Learning/Deep Learning architectures that can be ported to low-end devices, such as microcontrollers (Arduino Nano BLE 33, ESP32, STM32, etc.), which have less computational power than high-end devices such as single-board computers (Raspberry Pi Zero/3/4/5, NVIDIA Jetson, etc.). Both classes will allow the implementation of such AI architectures, defining a broader area recently named as Tiny Machine Learning (TinyML). This novel area has been consolidating itself in the literature with works applied to the domains of smart cities, medical assistance, agriculture, military, autonomous robotics, among many others. Such possibilities are mainly due to the development of model optimisation techniques, such as Quantisation and Pruning, and the evolution of wireless network technologies that allow the transmission of such optimised models from the cloud environment, where they were trained, validated, optimised, converted, and stored, toward IoIT devices spread across multiple locations, allowing them to update their decision-making functions, completing the Tiny Machine Learning Operations (TinyMLOps) cycle.

This article surveyed research works in this comprehensive area, finding and classifying a number of very recent contributions focused on the development, analysis, and testing of TinyML models on low and high-end embedded platforms through frameworks such as TensorFlow Lite, Keras, TensorFlow Lite Micro, and Edge Impulse. Moreover, we identified challenges that intersect the three convergent areas that define the Internet of Intelligent Things paradigm, encompassing issues that are related to its distributed nature (with devices spread across multiple locations and contexts) and resource constraints. Although TinyML is not necessarily IoIT, since there are many other issues to consider, this article discussed how compressed ML/DL models deployed on embedded devices will be the central element of Internet of Intelligent Things.

Overall, it was discussed that the identified key challenges predominantly arose from the resource limitations of IoIT edge devices. These constraints pose significant obstacles to deploying more sophisticated ML/DL architectures that necessitate greater computational capabilities. It could be seen that some current research efforts have been centred on the investigation of metrics most linked with model correctness and their packaging for implementation on low and high-end systems.

Furthermore, we discovered a research gap in the analysis of how existing models work on platforms when applied to the Internet of Intelligent Things context, which necessitates smart devices on the edge that consume low energy and that are affordable. When bringing ML/DL models to the edge, such concerns should foster new research efforts to create a more adequate hardware landscape, supporting further developments of IoIT-based applications.

Finally, after achieving our expected goals in this survey article, future works will be dedicated to the development of extensive performance benchmarking, taking into account a variety of popular low-end and high-end platforms, as well as different ML/DL models from different application contexts. This is expected to support the understanding, through metrics on performance and computational resource consumption, about the configurations that would better allow the implementation of IoIT devices that meet the three criteria defined in this work. Doing so, we want to further support the analysis of IoIT solutions, paving the way for more transformative developments in this area.

CRediT authorship contribution statement

Franklin Oliveira: Writing – review & editing, Writing – original draft, Investigation, Conceptualization. **Daniel G. Costa:** Writing – review & editing, Writing – original draft, Supervision, Project administration, Methodology. **Flávio Assis:** Writing – original draft, Supervision. **Ivanovitch Silva:** Writing – review & editing, Writing – original draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgment

This work was financially supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

[1] M. Weiser, The computer for the 21st century: specialized elements of hardware and software, connected by wires, radio waves and infrared, will be so ubiquitous that no one will notice their presence, in: Readings in Human–Computer Interaction, Elsevier, 1995, pp. 933–940.

[2] P. Marwedel, Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems, and the Internet of Things, Springer Nature, 2021.

[3] K.A. Aldahdouh, K.A. Darabkh, W. Al-Sit, et al., A survey of 5G emerging wireless technologies featuring LoRaWAN, sigfox, NB-IoT and LTE-M, in: 2019 International Conference on Wireless Communications Signal Processing and Networking, WiSPNET, IEEE, 2019, pp. 561–566.

[4] F. Firouzi, K. Chakrabarty, S. Nassif, Intelligent Internet of Things: From Device to Fog and Cloud, Springer, 2020.

[5] H. Landaluce, L. Arjona, A. Perallos, F. Falcone, I. Angulo, F. Muralter, A review of IoT sensing applications and challenges using RFID and wireless sensor networks, Sensors 20 (9) (2020) 2495, http://dx.doi.org/10.3390/s20092495.

[6] T. Domínguez-Bolaño, O. Campos, V. Barral, C.J. Escudero, J.A. García-Naya, An overview of IoT architectures, technologies, and existing open-source projects, Internet Things 20 (2022) 100626, http://dx.doi.org/10.1016/j.iot.2022.100626.

[7] K. Cao, Y. Liu, G. Meng, Q. Sun, An overview on edge computing research, IEEE Access 8 (2020) 85714–85728, http://dx.doi.org/10.1109/ACCESS.2020.2991734.

[8] M. De Donno, K. Tange, N. Dragoni, Foundations and evolution of modern computing paradigms: Cloud, iot, edge, and fog, IEEE Access 7 (2019) 150936–150948.

[9] M. Hussain, L.-F. Wei, A. Lakhan, S. Wali, S. Ali, A. Hussain, Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing, Sustain. Comput. Inform. Syst. 30 (2021) 100517.

[10] C. Zhang, Intelligent internet of things service based on artificial intelligence technology, in: 2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering, ICBAIE, 2021, pp. 731–734, http://dx.doi.org/10.1109/ICBAIE52039.2021.9390061.

[11] K. Shafique, B.A. Khawaja, F. Sabir, S. Qazi, M. Mustaqim, Internet of Things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT scenarios, IEEE Access 8 (2020) 23022–23040, http://dx.doi.org/10.1109/ACCESS.2020.2970118.

[12] P. Andrade, I. Silva, M. Diniz, T. Flores, D.G. Costa, E. Soares, Online processing of vehicular data on the edge through an unsupervised TinyML regression technique, ACM Trans. Embed. Comput. Syst. (2023) http://dx.doi.org/10.1145/3591356.

[13] P. Andrade, I. Silva, M. Silva, T. Flores, J. Cassiano, D.G. Costa, A tinyml soft-sensor approach for low-cost detection and monitoring of vehicular emissions, Sensors 22 (10) (2022) 3838, http://dx.doi.org/10.3390/s22103838.

[14] K. Fang, Z. Xu, Y. Li, J. Pan, A fall detection using sound technology based on TinyML, in: 2021 11th International Conference on Information Technology in Medicine and Education, ITME, IEEE, 2021, pp. 222–225.

[15] M. Antonini, M. Pincheira, M. Vecchio, F. Antonelli, An adaptable and unsupervised TinyML anomaly detection system for extreme industrial environments, Sensors 23 (4) (2023) 2344, http://dx.doi.org/10.3390/s23042344.

[16] A. Mellit, An embedded solution for fault detection and diagnosis of photovoltaic modules using thermographic images and deep convolutional neural networks, Eng. Appl. Artif. Intell. 116 (2022) 105459.

[17] S.F. Ahmed, M.S.B. Alam, M. Hoque, A. Lameesa, S. Afrin, T. Farah, M. Kabir, G. Shafiullah, S. Muyeen, Industrial Internet of Things enabled technologies, challenges, and future directions, Comput. Electr. Eng. 110 (2023) 108847, http://dx.doi.org/10.1016/j.compeleceng.2023.108847.

[18] N. Schizas, A. Karras, C. Karras, S. Sioutas, TinyML for ultra-low power AI and large scale IoT deployments: A systematic review, Future Internet 14 (12) (2022) 363, http://dx.doi.org/10.3390/fi14120363.

[19] S.S. Saha, S.S. Sandha, M. Srivastava, Machine learning for microcontroller-class hardware: A review, IEEE Sens. J. 22 (22) (2022) 21362–21390, http://dx.doi.org/10.1109/JSEN.2022.3210773.

[20] L. Dutta, S. Bharali, Tinyml meets iot: A comprehensive survey, Internet Things 16 (2021) 100461, http://dx.doi.org/10.1016/j.iot.2021.100461.

[21] Y. Qian, D. Wu, W. Bao, P. Lorenz, The Internet of Things for smart cities: Technologies and applications, IEEE Netw. 33 (2) (2019) 4–5.

[22] M. Mattioli, The apollo guidance computer, IEEE Micro 41 (6) (2021) 179–182.

[23] K.R. Raghunathan, History of microcontrollers: First 50 years, IEEE Micro 41 (6) (2021) 97–104.

[24] J. Yick, B. Mukherjee, D. Ghosal, Wireless sensor network survey, Comput. Netw. 52 (12) (2008) 2292–2330.

[25] A.W. Bhat, A. Passi, Wireless sensor network motes: A comparative study, in: 2022 9th International Conference on Computing for Sustainable Global Development, INDIACom, IEEE, 2022, pp. 141–144.

[26] M. Johnson, M. Healy, P. Van de Ven, M.J. Hayes, J. Nelson, T. Newe, E. Lewis, A comparative review of wireless sensor network mote technologies, SENSORS, 2009 IEEE (2009) 1439–1442.

[27] S. Mittal, A survey on optimized implementation of deep learning models on the nvidia jetson platform, J. Syst. Archit. 97 (2019) 428–442.

[28] Z. Islam, P. Kim Cheng Low, I. Hasan, Intention to use advanced mobile phone services (AMPS), Manag. Decis. 51 (4) (2013) 824–838.

[29] P. Sharma, Evolution of mobile wireless communication networks-1G to 5G as well as future prospective of next generation communication network, Int. J. Comput. Sci. Mob. Comput. 2 (8) (2013) 47–53.

[30] L.J. Vora, Evolution of mobile generation technology: 1G to 5G and review of upcoming wireless technology 5G, Int. J. Mod. Trends Eng. Res. 2 (10) (2015) 281–290.

[31] C.M. Ramya, M. Shanmugaraj, R. Prabakaran, Study on ZigBee technology, in: 2011 3rd International Conference on Electronics Computer Technology, Vol. 6, IEEE, 2011, pp. 297–301.

[32] M. Kocakulak, I. Butun, An overview of wireless sensor networks towards Internet of Things, in: 2017 IEEE 7th Annual Computing and Communication Workshop and Conference, CCWC, Ieee, 2017, pp. 1–6.

[33] R.S. Sinha, Y. Wei, S.-H. Hwang, A survey on LPWA technology: LoRa and NB-IoT, Ict Express 3 (1) (2017) 14–21.

[34] N. Hassan, K.-L.A. Yau, C. Wu, Edge computing in 5G: A review, IEEE Access 7 (2019) 127276–127289.

[35] A.T. Jawad, R. Maaloul, L. Chaari, A comprehensive survey on 6G and beyond: Enabling technologies, opportunities of machine learning and challenges, Comput. Netw. 237 (2023) 110085, http://dx.doi.org/10.1016/j.comnet.2023.110085.

[36] A. Alotaibi, A. Barnawi, Securing massive IoT in 6G: Recent solutions, architectures, future directions, Internet Things 22 (2023) 100715, http://dx.doi.org/10.1016/j.iot.2023.100715.

[37] H. Al-Sahaf, Y. Bi, Q. Chen, A. Lensen, Y. Mei, Y. Sun, B. Tran, B. Xue, M. Zhang, A survey on evolutionary machine learning, J. Royal Soc. New Zealand 49 (2) (2019) 205–228.

[38] M. Alloghani, D. Al-Jumeily, J. Mustafina, A. Hussain, A.J. Aljaaf, A systematic review on supervised and unsupervised machine learning algorithms for data science, in: Supervised and Unsupervised Learning for Data Science, Springer, 2020, pp. 3–21.

[39] A.E. Ezugwu, A.M. Ikotun, O.O. Oyelade, L. Abualigah, J.O. Agushaka, C.I. Eke, A.A. Akinyelu, A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects, Eng. Appl. Artif. Intell. 110 (2022) 104743, http://dx.doi.org/10.1016/j.engappai.2022.104743.

[40] S. Kamm, S.S. Veekati, T. Müller, N. Jazdi, M. Weyrich, A survey on machine learning based analysis of heterogeneous data in industrial automation, Comput. Ind. 149 (2023) 103930, http://dx.doi.org/10.1016/j.compind.2023.103930.

[41] L.V. Haar, T. Elvira, O. Ochoa, An analysis of explainability methods for convolutional neural networks, Eng. Appl. Artif. Intell. 117 (2023) 105606, http://dx.doi.org/10.1016/j.engappai.2022.105606.

[42] S. Dong, P. Wang, K. Abbas, A survey on deep learning and its applications, Comp. Sci. Rev. 40 (2021) 100379.

[43] X. Feng, Y. Jiang, X. Yang, M. Du, X. Li, Computer vision algorithms and hardware implementations: A survey, Integration 69 (2019) 309–320, http://dx.doi.org/10.1016/j.vlsi.2019.07.005.

[44] M.A. khelili, S. slatnia, O. kazar, A. merizig, S. mirjalili, Deep learning and metaheuristics application in Internet of Things: A literature review, Microprocess. Microsyst. 98 (2023) 104792, http://dx.doi.org/10.1016/j.micpro.2023.104792.

[45] A.L. Fradkov, Early history of machine learning, IFAC-PapersOnLine 53 (2) (2020) 1385–1390.

[46] S. Dargan, M. Kumar, M.R. Ayyagari, G. Kumar, A survey of deep learning and its applications: a new paradigm to machine learning, Arch. Comput. Methods Eng. 27 (2020) 1071–1092.

[47] A. Aggarwal, M. Mittal, G. Battineni, Generative adversarial network: An overview of theory and applications, Int. J. Inf. Manag. Data Insights 1 (1) (2021) 100004.

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).

[49] W.X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, et al., A survey of large language models, 2023, arXiv preprint arXiv:2303.18223.

[50] S. Khan, M. Naseer, M. Hayat, S.W. Zamir, F.S. Khan, M. Shah, Transformers in vision: A survey, ACM Comput. Surv. 54 (10s) (2022) 1–41.

[51] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A.C. Berg, W.-Y. Lo, et al., Segment anything, 2023, arXiv preprint arXiv:2304.02643.

[52] C. MacKay, W. Klement, P. Vanberkel, N. Lamond, R. Urquhart, M. Rigby, A framework for implementing machine learning in healthcare based on the concepts of preconditions and postconditions, Healthc. Anal. 3 (2023) 100155, http://dx.doi.org/10.1016/j.health.2023.100155.

[53] İ. Yazici, I. Shayea, J. Din, A survey of applications of artificial intelligence and machine learning in future mobile networks-enabled systems, Eng. Sci. Technol. Int. J. 44 (2023) 101455, http://dx.doi.org/10.1016/j.jestch.2023.101455.

[54] R. Bianchini, M. Fontoura, E. Cortez, A. Bonde, A. Muzio, A.-M. Constantin, T. Moscibroda, G. Magalhaes, G. Bablani, M. Russinovich, Toward ml-centric cloud platforms, Commun. ACM 63 (2) (2020) 50–59, http://dx.doi.org/10.1145/3364684.

[55] N.N. Alajlan, D.M. Ibrahim, Tinyml: Enabling of inference deep learning models on ultra-low-power IoT edge devices for AI applications, Micromachines 13 (6) (2022) http://dx.doi.org/10.3390/mi13060851.

[56] M. Shafique, T. Theocharides, V.J. Reddy, B. Murmann, Tinyml: current progress, research challenges, and future roadmap, in: 2021 58th ACM/IEEE Design Automation Conference, DAC, IEEE, 2021, pp. 1303–1306.

[57] H. Han, J. Siebert, TinyML: A systematic review and synthesis of existing research, in: 2022 International Conference on Artificial Intelligence in Information and Communication, ICAIIC, IEEE, 2022, pp. 269–274.

[58] D.G. Costa, C. Duran-Faundez, Open-source electronics platforms as enabling technologies for smart cities: Recent developments and perspectives, Electronics 7 (12) (2018) http://dx.doi.org/10.3390/electronics7120404.

[59] J.A. Ariza, H. Baez, Understanding the role of single-board computers in engineering and computer science education: A systematic literature review, Comput. Appl. Eng. Educ. 30 (1) (2022) 304–329.

[60] D.G. Costa, F.P. de Oliveira, A prioritization approach for optimization of multiple concurrent sensing applications in smart cities, Future Gener. Comput. Syst. 108 (2020) 228–243, http://dx.doi.org/10.1016/j.future.2020.02.067.

[61] M.S. Rahman, T. Ghosh, N.F. Aurna, M.S. Kaiser, M. Anannya, A.S. Hosen, Machine learning and Internet of Things in industry 4.0: A review, Measur. Sensors 28 (2023) 100822, http://dx.doi.org/10.1016/j.measen.2023.100822.

[62] J.A. Ariza, H. Baez, Understanding the role of single-board computers in engineering and computer science education: A systematic literature review, Comput. Appl. Eng. Educ. 30 (1) (2022) 304–329.

[63] W.A. Salah, B.A. Zneid, Evolution of microcontroller-based remote monitoring system applications, Int. J. Electr. Comput. Eng. 9 (4) (2019) 2354.

[64] S.A.R. Zaidi, A.M. Hayajneh, M. Hafeez, Q.Z. Ahmed, Unlocking edge intelligence through tiny machine learning (TinyML), IEEE Access 10 (2022) 100867–100877, http://dx.doi.org/10.1109/ACCESS.2022.3207200.

[65] R. Chéour, S. Khriji, M. abid, O. Kanoun, Microcontrollers for IoT: Optimizations, computing paradigms, and future directions, in: 2020 IEEE 6th World Forum on Internet of Things, WF-IoT, 2020, pp. 1–7, http://dx.doi.org/10.1109/WF-IoT48130.2020.9221219.

[66] B. Sudharsan, S. Salerno, D.-D. Nguyen, M. Yahya, A. Wahid, P. Yadav, J.G. Breslin, M.I. Ali, TinyML benchmark: Executing fully connected neural networks on commodity microcontrollers, in: 2021 IEEE 7th World Forum on Internet of Things, WF-IoT, 2021, pp. 883–884, http://dx.doi.org/10.1109/WF-IoT51360.2021.9595024.

[67] B. Sudharsan, J.G. Breslin, M.I. Ali, Edge2train: A framework to train machine learning models (svms) on resource-constrained iot edge devices, in: Proceedings of the 10th International Conference on the Internet of Things, 2020, pp. 1–8.

[68] M. Bojarski, C. Chen, J. Daw, A. Değirmenci, J. Deri, B. Firner, B. Flepp, S. Gogri, J. Hong, L. Jackel, et al., The NVIDIA pilotnet experiments, 2020, arXiv preprint arXiv:2010.08776.

[69] D. Kim, J. Cha, S. Oh, J. Jeong, AnoGAN-based anomaly filtering for intelligent edge device in smart factory, in: 2021 15th International Conference on Ubiquitous Information Management and Communication, IMCOM, IEEE, 2021, pp. 1–6.

[70] H. Ren, D. Anicic, T.A. Runkler, Tinyol: Tinyml with online-learning on microcontrollers, in: 2021 International Joint Conference on Neural Networks, IJCNN, IEEE, 2021, pp. 1–8.

[71] D.A. Pham, A.D. Le, D.T. Pham, H.B. Vo, Alerttrap: On designing an edge-computing remote insect monitoring system, in: 2021 8th NAFOSTED Conference on Information and Computer Science, NICS, IEEE, 2021, pp. 323–328.

[72] P. Andrade, I. Silva, G. Signoretti, M. Silva, J. Dias, L. Marques, D.G. Costa, An unsupervised tinyml approach applied for pavement anomalies detection under the internet of intelligent vehicles, in: 2021 IEEE International Workshop on Metrology for Industry 4.0 & IoT, MetroInd4. 0&IoT, IEEE, 2021, pp. 642–647.

[73] B. Sudharsan, P. Yadav, J.G. Breslin, M.I. Ali, Train++: An incremental ml model training algorithm to create self-learning iot devices, in: 2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation, SmartWorld/SCALCOM/UIC/ATC/IOP/SCI, IEEE, 2021, pp. 97–106.

[74] A.N. Roshan, B. Gokulapriyan, C. Siddarth, P. Kokil, Adaptive traffic control with TinyML, in: 2021 Sixth International Conference on Wireless Communications, Signal Processing and Networking, WiSPNET, IEEE, 2021, pp. 451–455.

[75] I. Saradopoulos, I. Potamitis, S. Ntalampiras, A.I. Konstantaras, E.N. Antonidakis, Edge computing for vision-based, urban-insects traps in the context of smart cities, Sensors 22 (5) (2022) 2006, http://dx.doi.org/10.3390/s22052006.

[76] Z.-D. Zhang, M.-L. Tan, Z.-C. Lan, H.-C. Liu, L. Pei, W.-X. Yu, CDNet: A real-time and robust crosswalk detection network on Jetson nano based on YOLOv5, Neural Comput. Appl. 34 (13) (2022) 10719–10730.

[77] Ö. Gültekin, E. Cinar, K. Özkan, A. Yazıcı, Real-time fault detection and condition monitoring for industrial autonomous transfer vehicles utilizing edge artificial intelligence, Sensors 22 (9) (2022) http://dx.doi.org/10.3390/s22093208.

[78] M.F. Alati, G. Fortino, J. Morales, J.M. Cecilia, P. Manzoni, Time series analysis for temperature forecasting using TinyML, in: 2022 IEEE 19th Annual Consumer Communications & Networking Conference, CCNC, IEEE, 2022, pp. 691–694.

[79] M. Bechtel, Q. Weng, H. Yun, DeepPicarMicro: Applying TinyML to autonomous cyber physical systems, in: 2022 IEEE 28th International Conference on Embedded and Real-Time Computing Systems and Applications, RTCSA, IEEE, 2022, pp. 120–127.

[80] A. Rana, Y. Dhiman, R. Anand, Cough detection system using TinyML, in: 2022 International Conference on Computing, Communication and Power Technology, IC3P, IEEE, 2022, pp. 119–122.

[81] K. Xu, H. Zhang, Y. Li, Y. Zhang, R. Lai, Y. Liu, An ultra-low power tinyml system for real-time visual processing at edge, IEEE Trans. Circuits Syst. II (2023).

[82] E. Kim, J. Kim, J. Park, H. Ko, Y. Kyung, TinyML-based classification in an ECG monitoring embedded system, Comput. Mater. Contin. 75 (1) (2023) 1751–1764.

[83] N.N. Alajlan, D.M. Ibrahim, DDD TinyML: A TinyML-based driver drowsiness detection model using deep learning, Sensors 23 (12) (2023) 5696, http://dx.doi.org/10.3390/s23125696.

[84] R. Srinivasagan, M. Mohammed, A. Alzahrani, TinyML-sensor for shelf life estimation of fresh date fruits, Sensors 23 (16) (2023) 7081, http://dx.doi.org/10.3390/s23167081.

[85] D. Avellaneda, D. Mendez, G. Fortino, A TinyML deep learning approach for indoor tracking of assets, Sensors 23 (3) (2023) 1542, http://dx.doi.org/10.3390/s23031542.

[86] V. Tsoukas, A. Gkogkidis, E. Boumpa, S. Papafotikas, A. Kakarountas, A gas leakage detection device based on the technology of TinyML, Technologies 11 (2) (2023) 45, http://dx.doi.org/10.3390/technologies11020045.

[87] U. Ulusoy, O. Eren, A. Demirhan, Development of an obstacle avoiding autonomous vehicle by using stereo depth estimation and artificial intelligence based semantic segmentation, Eng. Appl. Artif. Intell. 126 (2023) 106808, http://dx.doi.org/10.1016/j.engappai.2023.106808.

[88] M.B. Azevedo, T.d.A. de Medeiros, M.d.A. Medeiros, I. Silva, D.G. Costa, Detecting face masks through embedded machine learning algorithms: A transfer learning approach for affordable microcontrollers, Mach. Learn. Appl. (2023) 100498.

[89] A. Mellit, N. Blasuttigh, A.M. Pavan, TinyML for fault diagnosis of photovoltaic modules using edge impulse platform, in: 2023 11th International Conference on Smart Grid, IcSmartGrid, IEEE, 2023, pp. 01–05.

[90] I.N. Mihigo, M. Zennaro, A. Uwitonze, J. Rwigema, M. Rovai, On-device IoT-based predictive maintenance analytics model: Comparing TinyLSTM and TinyModel from edge impulse, Sensors 22 (14) (2022) http://dx.doi.org/10.3390/s22145174.

[91] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, D.O. Wu, Edge computing in industrial Internet of Things: Architecture, advances and challenges, IEEE Commun. Surv. Tutor. 22 (4) (2020) 2462–2488.

[92] M. Javaid, A. Haleem, R.P. Singh, R. Suman, Artificial intelligence applications for industry 4.0: A literature-based study, J. Ind. Integr. Manag. 7 (01) (2022) 83–111.

[93] R.R. Singh, S. Yash, S. Shubham, V. Indragandhi, V. Vijayakumar, P. Saravanan, V. Subramaniyaswamy, IoT embedded cloud-based intelligent power quality monitoring system for industrial drive application, Future Gener. Comput. Syst. 112 (2020) 884–898.

[94] D.A. Gzar, A.M. Mahmood, M.K.A. Al-Adilee, Recent trends of smart agricultural systems based on Internet of Things technology: A survey, Comput. Electr. Eng. 104 (2022) 108453, http://dx.doi.org/10.1016/j.compeleceng.2022.108453.

[95] A.R. Madushanki, M.N. Halgamuge, W.S. Wirasagoda, A. Syed, Adoption of the Internet of Things (IoT) in agriculture and smart farming towards urban greening: A review, Int. J. Adv. Comput. Sci. Appl. 10 (4) (2019) 11–28.

[96] L. Mishra, S. Varma, et al., Internet of things for military applications, in: 2020 7th International Conference on Computing for Sustainable Global Development, INDIACom, IEEE, 2020, pp. 118–123.

[97] J. Kua, S.W. Loke, C. Arora, N. Fernando, C. Ranaweera, Internet of Things in space: a review of opportunities and challenges from satellite-aided computing to digitally-enhanced space living, Sensors 21 (23) (2021) 8117, http://dx.doi.org/10.3390/s21238117.

[98] V. Tsoukas, E. Boumpa, G. Giannakas, A. Kakarountas, A review of machine learning and TinyML in healthcare, in: Proceedings of the 25th Pan-Hellenic Conference on Informatics, 2021, pp. 69–73.

[99] V. Tsoukas, A. Gkogkidis, A. Kakarountas, Internet of things challenges and the emerging technology of TinyML, in: 2023 19th International Conference on Distributed Computing in Smart Systems and the Internet of Things, DCOSS-IoT, IEEE, 2023, pp. 491–495.

[100] A. Mellit, N. Blasuttigh, A.M. Pavan, TinyML for fault diagnosis of photovoltaic modules using edge impulse platform, in: 2023 11th International Conference on Smart Grid, IcSmartGrid, IEEE, 2023, pp. 01–05.

[101] R. David, J. Duke, A. Jain, V. Janapa Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, T. Wang, et al., Tensorflow lite micro: Embedded machine learning for TinyML systems, Proc. Mach. Learn. Syst. 3 (2021) 800–811.

[102] M. Antonini, M. Pincheira, M. Vecchio, F. Antonelli, Tiny-MLOps: A framework for orchestrating ML applications at the far edge of IoT systems, in: 2022 IEEE International Conference on Evolving and Adaptive Intelligent Systems, EAIS, IEEE, 2022, pp. 1–8.

[103] F. Zhengxin, Y. Yi, Z. Jingyu, L. Yue, M. Yuechen, L. Qinghua, X. Xiwei, W. Jeff, W. Chen, Z. Shuai, et al., MLOps spanning whole machine learning life cycle: A survey, 2023, arXiv preprint arXiv:2304.07296.

[104] S. Alla, S.K. Adari, S. Alla, S.K. Adari, What is mlops? in: Beginning MLOps with MLFlow: Deploy Models in AWS SageMaker, Google Cloud, and Microsoft Azure, Springer, 2021, pp. 79–124.

[105] E. Raj, D. Buffoni, M. Westerlund, K. Ahola, Edge mlops: An automation framework for aiot applications, in: 2021 IEEE International Conference on Cloud Engineering, IC2E, IEEE, 2021, pp. 191–200.

[106] K. Paupamah, S. James, R. Klein, Quantisation and pruning for neural network compression and regularisation, in: 2020 International SAUPEC/RobMech/PRASA Conference, IEEE, 2020, pp. 1–6.

[107] R.-Y. Sun, Optimization for deep learning: An overview, J. Oper. Res. Soc. China 8 (2) (2020) 249–294.

[108] W. Chen, H. Qiu, J. Zhuang, C. Zhang, Y. Hu, Q. Lu, T. Wang, Y. Shi, M. Huang, X. Xu, Quantization of deep neural networks for accurate edge computing, ACM J. Emerg. Technol. Comput. Syst. (JETC) 17 (4) (2021) 1–11.

[109] R. David, J. Duke, A. Jain, V. Janapa Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, T. Wang, et al., Tensorflow lite micro: Embedded machine learning for TinyML systems, Proc. Mach. Learn. Syst. 3 (2021) 800–811.

[110] S. Ashfaq, M. AskariHemmat, S. Sah, E. Saboori, O. Mastropietro, A. Hoffman, Accelerating deep learning model inference on arm cpus with ultra-low bit quantization and runtime, 2022, arXiv preprint arXiv:2207.08820.

[111] W. Wei, S. Islam, J. Banerjee, S. Zhou, C. Pan, C. Ding, M. Xie, An intermittent OTA approach to update the DL weights on energy harvesting devices, in: 2022 23rd International Symposium on Quality Electronic Design, ISQED, IEEE, 2022, pp. 1–6.

[112] V. Rajapakse, I. Karunanayake, N. Ahmed, Intelligence at the extreme edge: A survey on reformable TinyML, ACM Comput. Surv. 55 (13s) (2023) 1–30.

[113] L. Yang, A. Shami, A lightweight concept drift detection and adaptation framework for IoT data streams, IEEE Internet Things Mag. 4 (2) (2021) 96–101.

[114] Q. Duan, J. Huang, S. Hu, R. Deng, Z. Lu, S. Yu, Combining federated learning and edge computing toward ubiquitous intelligence in 6G network: Challenges, recent advances, and future directions, IEEE Commun. Surv. Tutor. 25 (4) (2023) 2892–2950, http://dx.doi.org/10.1109/COMST.2023.3316615.

[115] Q. Duan, S. Hu, R. Deng, Z. Lu, Combined federated and split learning in edge computing for ubiquitous intelligence in Internet of Things: State-of-the-art and future directions, Sensors 22 (16) (2022) http://dx.doi.org/10.3390/s22165983.

[116] F. Bayram, B.S. Ahmed, A. Kassler, From concept drift to model degradation: An overview on performance-aware drift detectors, Knowl.-Based Syst. 245 (2022) 108632.

[117] G. Andresini, F. Pendlebury, F. Pierazzi, C. Loglisci, A. Appice, L. Cavallaro, Insomnia: Towards concept-drift robustness in network intrusion detection, in: Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security, 2021, pp. 111–122.

[118] D.M. Manias, I. Shaer, L. Yang, A. Shami, Concept drift detection in federated networked systems, in: 2021 IEEE Global Communications Conference, GLOBECOM, IEEE, 2021, pp. 1–6.

[119] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan, et al., Towards federated learning at scale: System design, Proc. Mach. Learn. Syst. 1 (2019) 374–388.

[120] M. Zhang, E. Wei, R. Berry, Faithful edge federated learning: Scalability and privacy, IEEE J. Sel. Areas Commun. 39 (12) (2021) 3790–3804.

[121] D. Huba, J. Nguyen, K. Malik, R. Zhu, M. Rabbat, A. Yousefpour, C.-J. Wu, H. Zhan, P. Ustinov, H. Srinivas, et al., Papaya: Practical, private, and scalable federated learning, Proc. Mach. Learn. Syst. 4 (2022) 814–832.

[122] M. Shafique, T. Theocharides, V.J. Reddy, B. Murmann, Tinyml: current progress, research challenges, and future roadmap, in: 2021 58th ACM/IEEE Design Automation Conference, DAC, IEEE, 2021, pp. 1303–1306.

[123] T.S. Ajani, A.L. Imoize, A.A. Atayero, An overview of machine learning within embedded and mobile devices–optimizations and applications, Sensors 21 (13) (2021) 4412, http://dx.doi.org/10.3390/s21134412.

[124] B. Sudharsan, S. Salerno, D.-D. Nguyen, M. Yahya, A. Wahid, P. Yadav, J.G. Breslin, M.I. Ali, Tinyml benchmark: Executing fully connected neural networks on commodity microcontrollers, in: 2021 IEEE 7th World Forum on Internet of Things, WF-IoT, IEEE, 2021, pp. 883–884.