Journal of
Low Power Electronics and Applications

MDPI

Review

MCU Intelligent Upgrades: An Overview of AI-Enabled Low-Power Technologies

Tong Zhang, Bosen Huang, Xiewen Liu, Jiaqi Fan, Junbo Li, Zhao Yue * and Yanfang Wang *

College of Electronic Information and Optical Engineering, Nankai University, Tianjin 300071, China; 2120240407@mail.nankai.edu.cn (T.Z.); 2311747@mail.nankai.edu.cn (B.H.); 2120230425@mail.nankai.edu.cn (X.L.); 2120240444@mail.nankai.edu.cn (J.F.); 2120240393@mail.nankai.edu.cn (J.L.)
* Correspondence: yuezhao@nankai.edu.cn (Z.Y.); wyanfang@nankai.edu.cn (Y.W.)

Abstract

Microcontroller units (MCUs) serve as the core components of embedded systems. In the era of smart IoT, embedded devices are increasingly deployed on mobile platforms, leading to a growing demand for low-power consumption. As a result, low-power technology for MCUs has become increasingly critical. This paper systematically reviews the development history and current technical challenges of MCU low-power technology. It then focuses on analyzing system-level low-power optimization pathways for integrating MCUs with artificial intelligence (AI) technology, including lightweight AI algorithm design, model pruning, AI acceleration hardware (NPU, GPU), and heterogeneous computing architectures. It further elaborates on how AI technology empowers MCUs to achieve comprehensive low power consumption from four dimensions: task scheduling, power management, inference engine optimization, and communication and data processing. Through practical application cases in multiple fields such as smart home, healthcare, industrial automation, and smart agriculture, it verifies the significant advantages of MCUs combined with AI in performance improvement and power consumption optimization. Finally, this paper focuses on the key challenges that still need to be addressed in the intelligent upgrade of future MCU low power consumption and proposes in-depth research directions in areas such as the balance between lightweight model accuracy and robustness, the consistency and stability of edge-side collaborative computing, and the reliability and power consumption control of the sensor-storage-computing integrated architecture, providing clear guidance and prospects for future research.

Keywords: microcontroller units; embedded systems; AI; intelligent upgrades; low power consumption

check for updates

Academic Editor: Amit Kumar Singh

Received: 28 July 2025
Revised: 11 September 2025
Accepted: 25 September 2025
Published: 1 October 2025

Citation: Zhang, T.; Huang, B.; Liu, X.; Fan, J.; Li, J.; Yue, Z.; Wang, Y. MCU Intelligent Upgrades: An Overview of AI-Enabled Low-Power Technologies. *J. Low Power Electron. Appl.* **2025**, *15*, 60. https://doi.org/10.3390/jlpea15040060

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

1.1. The Evolution of Low-Power Requirements for MCUs

Against the backdrop of rapid development in embedded systems, MCUs, as the core components and control units of embedded systems, are undergoing a paradigm shift from basic logic control to intelligent decision making. Their applications span a wide range of fields, including industrial control, automotive electronics, consumer electronics, and the Internet of Things (IoT). With the rapid development of emerging fields such as smart IoT, Industry 4.0, and new energy vehicles, application scenarios continue to expand [1–4]. Low power consumption, as one of the key performance metrics for MCUs, can be analyzed

from the perspective of technological evolution and divided into three critical stages, as shown in Table 1.

Table 1. Three stages of evolution in low-power requirements for MCUs.

Stage	Key Scenes	Key Technology	Popular MCU
Stage 1	Traditional industrial equipment, low-end home appliances	Reduce static current	8-bit
Stage 2	Wearable devices, IoT sensors	Sleep–wake mechanism, DVFS technology	16-bit, 32-bit
Stage 3	Smart IoT, Industry 4.0, New Energy Vehicles	System architecture optimization, AI integration	32-bit

The first stage only has basic drive functions, with the primary focus on static power consumption optimization. The main role of the MCU is to perform simple logic control and data acquisition, and it is used in traditional industrial equipment and lower-end electronic products, such as the control of household appliances in everyday life [5]. Devices in this stage mostly use wired power supply or high-capacity batteries, with relatively relaxed power consumption requirements. Low-power technology focuses on reducing static current to improve basic battery life. For example, 8-bit MCUs have become the market mainstream due to their low cost and low power consumption. Yang et al. [6] proposed an isolated word speech recognition system-on-chip, which adopted an 8-bit MCU for its low cost and low power consumption advantages. However, dynamic power consumption control has not yet received sufficient attention in this stage, and low-power technology remains in its early stages.

The second phase gradually expanded into mobile scenarios, introducing dynamic power consumption control. With the rise of mobile devices such as smartphones and wearable devices, MCUs have been widely adopted in scenarios with higher battery life requirements, driving low-power technology from “static optimization” to “dynamic optimization.” The key technological advancement in this phase was the introduction of DVFS (Dynamic Voltage and Frequency Scaling) [7] technology, enabling MCUs to simultaneously reduce voltage and frequency across different operating states, thereby controlling overall power consumption; sleep–wake mechanisms became mainstream, reducing energy consumption during long-term operation through low-power standby and event-driven wake-up. Typical applications include the low-power wireless sensor network SoC (System on Chip) designed by Tiwari A et al. [8]; the maturation of embedded flash memory and low-leakage design technologies further enhanced MCU energy efficiency [9]. During this phase, 16-bit and 32-bit MCUs gradually replaced 8-bit MCUs as the mainstream, meeting the demand for performance upgrades while enabling more granular dynamic power consumption control through software-hardware collaboration [10,11].

In the third phase, with the explosive development of AI, intelligent requirements have been placed on MCUs in scenarios such as smart IoT, Industry 4.0, and new energy vehicles. MCU low-power technology has entered a new era centered on system-level optimization and edge intelligence collaboration. Due to the need for edge devices to process complex perception tasks (such as image recognition and voice interaction), MCUs are facing unprecedented challenges in terms of computing power and power consumption. By integrating heterogeneous computing architectures, lightweight neural network models, and edge AI acceleration units (such as NPUs and DSPs), MCUs can achieve high-

performance, low-power data inference [12]. Additionally, the overall system architecture must incorporate strategies such as task scheduling, data caching, and energy efficiency grading to drive MCUs from instruction-level power optimization toward full-stack power control [13]. Furthermore, demands for device lifespan, thermal management, and remote deployment capabilities have compelled MCUs to incorporate more complex power management and security isolation mechanisms into low-power designs [14]. At this stage, 32-bit MCUs have achieved widespread adoption, and low-power technology is no longer confined to device-level optimizations but is evolving toward cross-level system-level collaborative design, forming an intelligent low-power technology roadmap driven by the joint integration of "architecture-algorithm-task-scenario."

Despite significant progress in low-power technology, current MCUs still face numerous challenges in the low-power domain. In IoT scenarios, multimodal data poses challenges to MCUs in terms of preprocessing, feature extraction, and model training capabilities. Multimodal data processing imposes stringent requirements on MCU computing power. Traditional MCUs suffer from obvious bottlenecks in terms of insufficient floating-point computing capabilities, limited interface bandwidth, and inadequate memory capacity, making it difficult to efficiently process high-frequency real-time data and limiting real-time data analysis capabilities in low-power environments [15–17].

As AI models are increasingly deployed on edge computing devices, the conflict between the power consumption requirements of model computations and the low-power requirements of edge devices is growing increasingly pronounced. On one hand, the complexity of AI inference is escalating, necessitating greater computational power; on the other hand, MCUs must maintain low power consumption to align with the energy specifications of the corresponding devices. In the actual deployment of AI models on MCUs, issues such as low computational resource utilization and high peak power consumption ratios ultimately impact the system's overall efficiency and power consumption [18]. Additionally, the "memory wall" problem caused by external storage devices failing to keep pace with computational units [19] also affects the system's overall stability and energy efficiency. In most traditional systems, computational units and memory modules are separated, and the repeated data transfers between them are the fundamental cause of latency and high power consumption. Compared to computational units, accessing memory is less energy-efficient. However, the current low reuse rate of data input to AI models exacerbates this issue. Existing MCU power management primarily involves static optimization and has not yet established a system-level dynamic power coordination control mechanism [20].

MCUs are used in fields such as healthcare, finance, and industry, and require robust security protection measures. However, the introduction of comprehensive protection mechanisms often results in additional resource and power consumption overhead, which directly conflicts with low-power objectives [21]. Most protection mechanisms focus only on a single stage or module, neglecting other stages, which allows attackers to obtain sensitive information from a single stage [22]. Comprehensive protection mechanisms should cover the entire process from data collection, transmission, storage, to processing. However, covering every stage typically comes with greater resource overhead, which may further increase MCU power consumption, conflicting with low-power objectives [23]. Therefore, how to enhance security protection while balancing low-power requirements has become a critical challenge in MCU design.

MCUs have evolved from simple logic control to complex intelligent functionality, and their low-power design has placed higher demands on the development process. However, the rapid improvement in system functionality has also led to an increase in development difficulty. Once development efficiency declines, the implementation of MCU low-power optimization and rapid product iteration will be directly affected. Driven

by the gap between developers' vision and domain requirements, the existence of low-code development platforms is crucial [24]. Currently, MCU development toolchains are fragmented and lack integration, with ineffective coordination between hardware and software platforms. In particular, debugging tools cannot perform a comprehensive analysis of computing power, power consumption, and real-time performance, making it difficult for developers to promptly identify and optimize high-power consumption areas within the system. Additionally, during the software development phase, there is a need to address the issue of deep integration between the operating system and intelligent algorithm frameworks while meeting the real-time requirements of intelligent inference and control tasks. However, the debugging process primarily relies on external tools, resulting in low efficiency [25]. The system verification phase faces challenges due to the broad scope of testing and insufficient automation levels, making it difficult to complete complex scenario testing in a short timeframe, significantly constraining the overall iteration speed [26].

To meet current application requirements, the low-power technology path for MCUs can be divided into multiple layers, as shown in Figure 1: In terms of process technology, advanced processes can be used to improve energy efficiency, as shown in Figure 1a, involving the NTV multi-core CPU [27]. In terms of architecture, heterogeneous multi-core designs can be adopted [28], and Figure 1b shows the proposed heterogeneous embedded software execution architecture based on multi-core cooperative operations. In terms of circuit design, optimized power management modules can be used to achieve multi-voltage control, and Figure 1c has a power analysis tool capable of measuring the power consumption of the ready queue structure mechanism for each implementation [29]. And at the software level, task scheduling and AI algorithms can be employed to reduce unnecessary power consumption [30], as shown in Figure 1d. These elements work in tandem to form the core support for reducing power consumption in the intelligent upgrading of MCUs. As analyzed above, traditional MCUs have significant shortcomings in terms of computing power, energy efficiency, security, and development efficiency. Therefore, innovative solutions that integrate AI with both software and hardware are needed to address these bottlenecks.

1.2. The Development of AI Technology and Hardware Integration

AI technology is evolving from traditional cloud-based centralized distribution to edge device-based distributed systems alongside the development of the Internet of Things (IoT) and embedded systems [31]. This evolution aligns perfectly with the demand for MCU intelligence, driving innovation in the integration of hardware and algorithms. The fusion of AI technology and hardware can be categorized into three aspects: algorithm adaptation to hardware, hardware adaptation to algorithms, and collaborative optimization of algorithms and hardware.

From an algorithmic perspective, traditional cloud-based deep learning frameworks are overly complex and resource-intensive, as they were originally designed for cloud-based GPUs (Graphics Processing Units). Attempting to deploy such frameworks on edge devices would result in excessive resource consumption, making direct deployment on MCUs impractical. For example, the peak memory requirements of deep neural network inference may far exceed the memory capacity of an MCU, rendering deployment impossible [32]. This requires us to lightweight the learning framework. First, through model compression, using operations such as Quantization and pruning, the model size is optimized to the KB level. Second, computational instructions are optimized for the MCU's internal central processing unit architecture to reduce redundant operations. Finally, static memory is used to replace dynamically allocated space, reducing memory fragmentation [33].

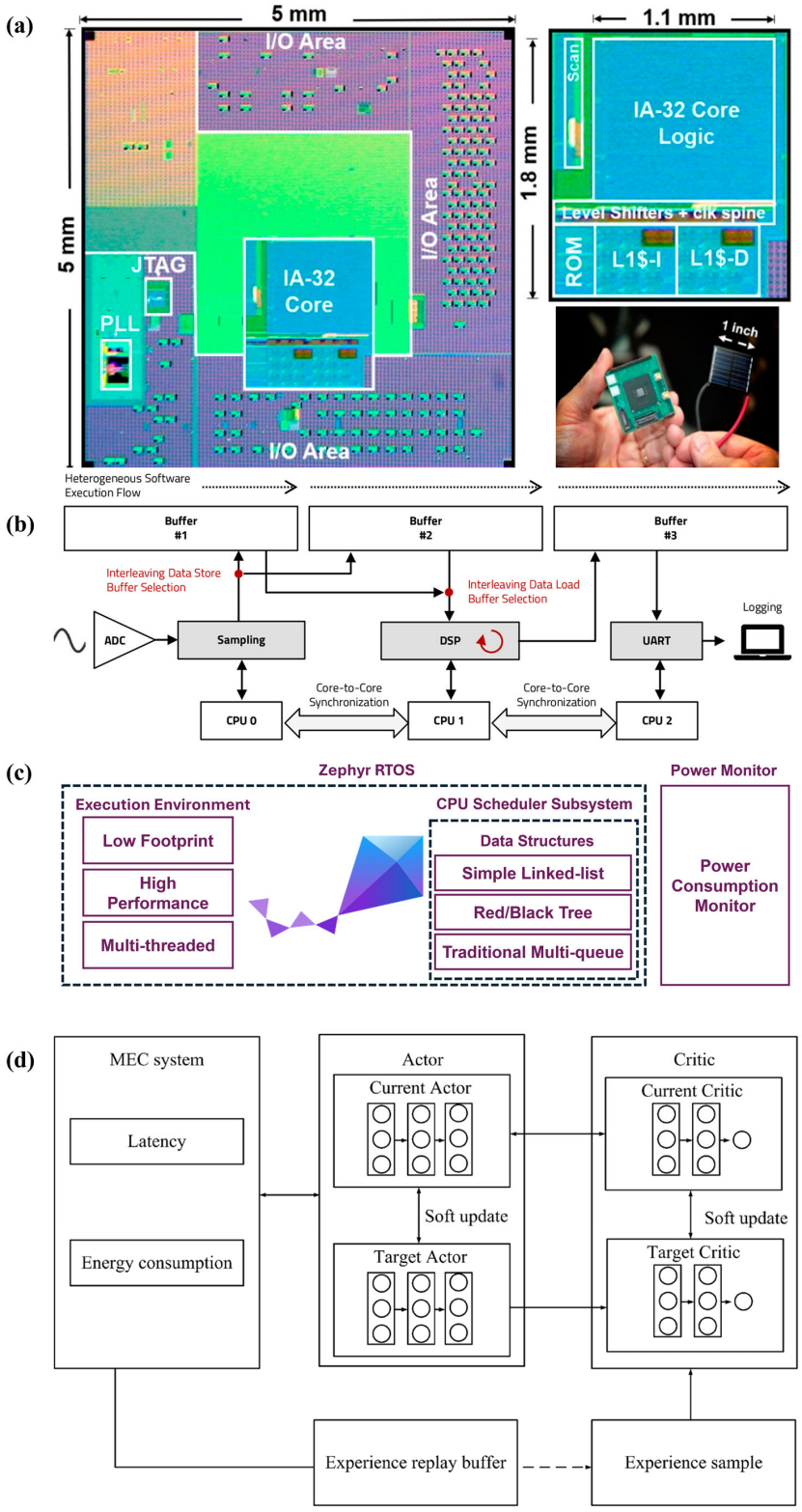


Figure 1. (**a**) NTV-CPU: Full-Chip 32 nm die and core micrographs and characteristics. Packaged IA-32 silicon and the small solar cell used to power the core [27]. (**b**) Proposal for a Heterogeneous Embedded Software Execution Architecture Based on Multi-Core Collaborative Operations [28]. (**c**) The proposed system consists of a hardware-based board running the Zephyr RTOS, modifications to the scheduler subsystem to support various ready queue structures, and a power profile tool used to measure power consumption for each implemented ready queue structure mechanism [29]. (**d**) The improved DDPG model architecture is employed for task scheduling in mobile edge computing systems, reducing latency and energy consumption [30].

At the hardware level, integrating AI accelerators into the MCU has become the core method for achieving intelligence. Due to the computational limitations of traditional MCUs, they are unable to handle complex AI tasks. To address this, there are two approaches to enhancing MCU computational capabilities: integrating an NPU (Neural Network Processing Unit) to specifically process models; and introducing vector extension instruction sets to enhance vector computing capabilities. Additionally, heterogeneous computing architectures can be used to dynamically allocate tasks, with the CPU handling logic, the NPU processing AI computations, and the DSP handling signals, among other functions. This allows different modules to handle different tasks, improving efficiency through data interaction.

As AI and hardware converge, diversification and low power consumption have become the primary development directions. However, low power consumption is the core constraint for MCUs integrating AI processing modules. The powerful inference capabilities of AI come with high power consumption issues, making it unsuitable for many scenarios such as wearable devices. To address power consumption issues, asynchronous circuit design can be adopted to avoid unnecessary power consumption caused by the global clock [34]; and adaptive computing power regulation [35] can be introduced to dynamically adjust the number of units participating in computations. As AI algorithms increasingly migrate to the edge, the intelligent upgrading of MCUs becomes both possible and necessary [36].

1.3. Intelligent Upgrade of Low-Power Technology for MCUs

The integration of MCU intelligence and AI edge computing is an inevitable choice driven by both technological advancements and market demands. We can utilize AI models to reduce MCU power consumption. While AI models themselves increase power consumption, due to computational requirements. System-level optimizations can significantly reduce unnecessary power consumption, far exceeding the increase caused by AI models. This enables the combination of MCU and AI to meet low-power requirements. Through edge computing, reliance on the cloud can be reduced, thereby lowering latency [37]. For example, in facial expression recognition during image processing, local deep learning inference can be used to determine facial structures, avoiding the latency caused by cloud transmission and reducing costs and power consumption. Edge preprocessing can filter redundant data, significantly reducing the transmission of invalid data during communication [38], thereby lowering communication bandwidth costs and transmission power consumption. It can also reduce the deployment of high-precision sensors, lowering hardware investment costs and overall power consumption, while also preventing data leaks during transmission. Data analysis is completed locally, with only abnormal results reported, thereby reducing the risk of data leaks [39]. Lightweight TEEs achieve code and data isolation in certain MCUs (e.g., ARM TrustZone) with low overhead; recent research indicates TEEs can provide isolated protection for critical task execution at edge nodes, making them suitable for resource-constrained environments [40].

In this paper, we have summarized the low-power technology routes for MCUs, discussed the current status and future trends of MCU-AI integration in light of the explosive development of AI models, and finally explored the application of MCU-AI integration in low-power technology, providing an overview of the future development of low-power technology for embedded systems. Figure 2 presents an overview diagram of the MCU + AI low-power technology, summarizing the approach outlined in this paper.

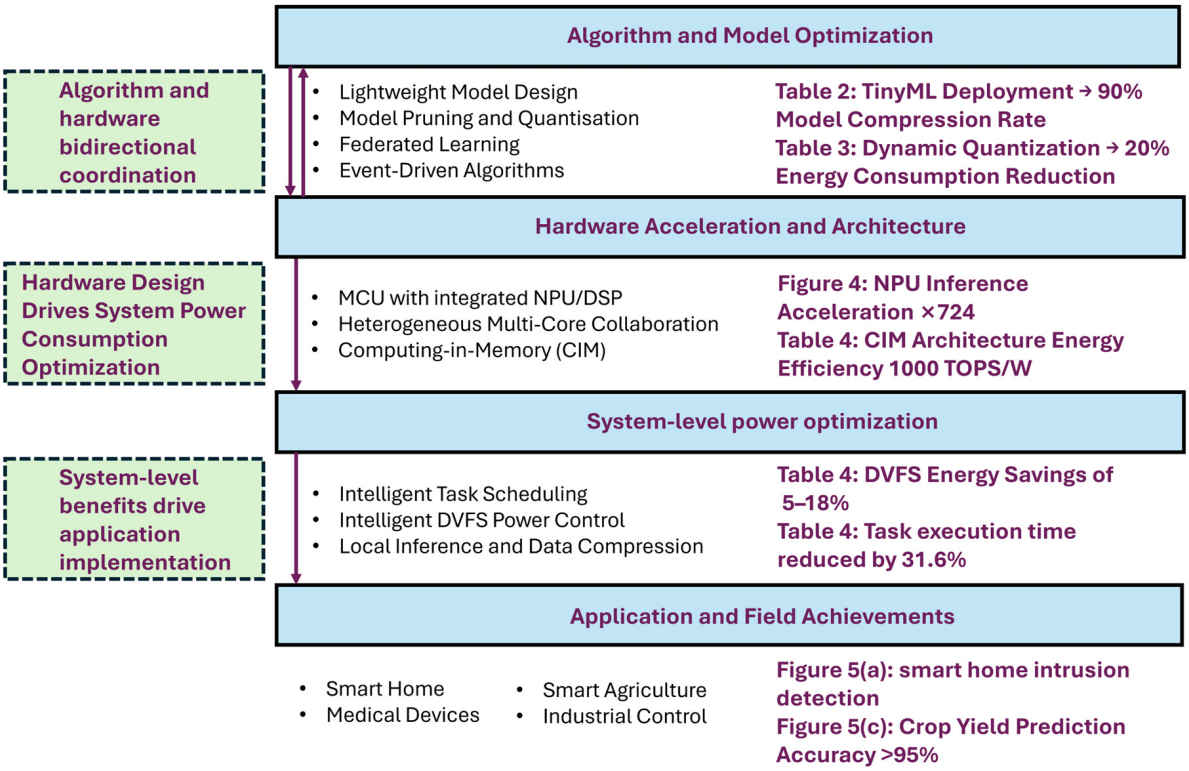


Figure 2. MCU + AI Low-Power Technology Overview Roadmap.

2. The Technical Approach of Combining MCU with AI

As AI algorithms are gradually deployed to the edge, efficiently integrating AI models into resource-constrained MCUs has become a key direction. The overall technical approach includes: adapting AI algorithms and models through lightweight optimization to align with MCU resource constraints, introducing AI acceleration units (such as NPUs or RISC-V architectures) to enhance computational power, and employing heterogeneous computing architectures and instruction set optimizations to balance performance and power consumption. The following sections detail the primary technical pathways for integrating MCUs with AI from both algorithmic and hardware perspectives.

2.1. Lightweight AI Algorithms and Model Adaptation

Complex and large AI algorithms cannot be deployed in MCUs with limited computing power, memory, and power consumption. Therefore, lightweight AI algorithms can be used to achieve efficient and reliable intelligent decision-making. In terms of lightweight design, we can achieve this by trimming and customizing the model architecture, optimizing model parameters, and changing the calculation method.

For embedded scenarios, we need to compress and prune traditional deep learning models. This helps develop highly compact and efficient neural network structures, which are then deployed on MCU platforms. For example, the TinyML technology shown in Figure 3 can balance model performance and resource consumption in such deployments. Samanta Riya et al. [41] proposed the TinyAerialNet model based on TinyML technology, enabling real-time, low-power, and low-cost onboard aerial image classification on resource-constrained MCUs, addressing the challenge of deploying traditional deep learning models on KB-level low-memory, low-storage devices. Kong et al. [42] introduced the EdgeCompress compression framework to reduce CNN computational overhead and eliminate redundant calculations on background regions.

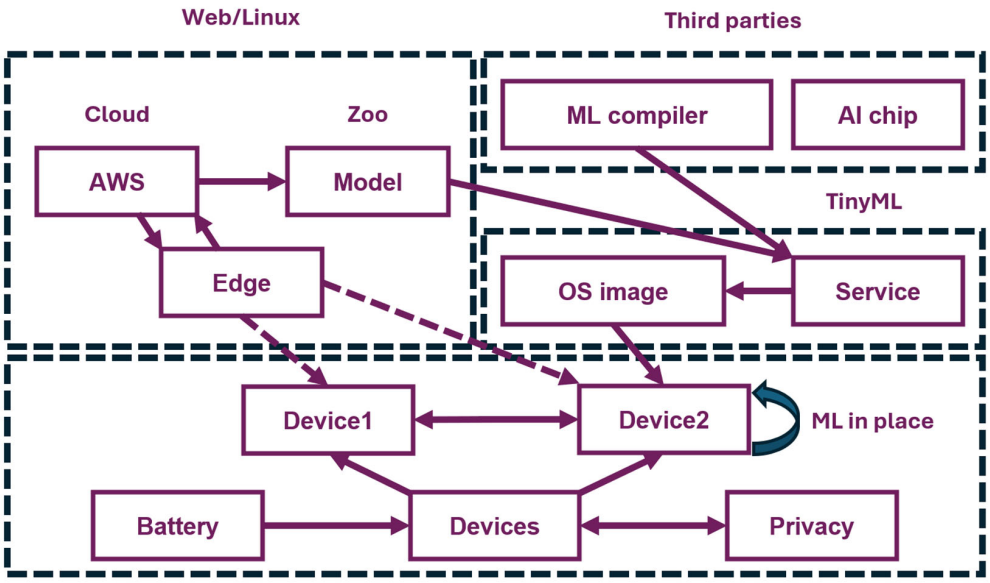


Figure 3. Shows the interaction between the three components: the compiler plugin interface, the orchestration protocol, and the inference module specification. Support for these three components indicates that IoT hardware can meet the basic requirements for implementing TinyML services.

Quantization and pruning of parameters within a model are currently the most widely applied and effective techniques. Table 2 and Table 3, respectively, present the MCU toolchain support table for Quantization and pruning, and the compression methods and toolchain support. Quantization technology significantly reduces model computational complexity and memory overhead by converting weights and activation values from 32-bit floating-point numbers to integer representations with lower bit widths. However, overly low Quantization precision may lead to a decline in model performance. Therefore, a dynamic Quantization strategy is introduced, which applies 8-bit Quantization to sensitive layers within the model and further compresses non-sensitive layers to 4-bit Quantization to achieve a balance between performance and efficiency. Song et al. [43] proposed a dynamic Quantization method for feature map sensitive regions, dividing the input feature map into sensitive and non-sensitive regions, performing INT8 convolution on sensitive regions, and INT4 convolution on non-sensitive regions. Liu et al. [44] proposed an innovative approach to dynamically allocate bit width for each layer based on the complexity of the input sample, using a lightweight bit controller to predict the optimal bit width sequence for each layer. Pruning techniques identify redundant parameters or connection structures in the model, which can be removed to simplify the model while maintaining inference efficiency. Xie et al. [45] designed an AI module based on ARM for ectopic heartbeat classification, using a lightweight convolutional neural network to address the limitations of traditional electrocardiogram monitoring devices. They pruned the structure of the traditional VGG19 model, reducing the number of parameters from 139.6 million to approximately 10,000, while keeping the accuracy decline within 2–5%.

Table 2. MCU Toolchain Quantization and Pruning Support.

Toolchain	Quantization Support	Pruning Support
LiteRT [46]	INT8 Quantization (Dynamic/Static) supports mixed precision (FP16/INT8)	Structured pruning (removing redundant channels/layers) combined with dynamic range Quantization to optimize model size

Table 2. *Cont.*

Toolchain	Quantization Support	Pruning Support
CMSIS-NN [47]	INT8/INT16 Quantization Q Format Conversion	Structural pruning must be manually implemented in conjunction with CMSIS-DSP, relying on the sparsity of quantized weights for optimization.
MicroTVM [48]	INT8 Quantization supports dynamic shapes	Structured pruning (based on TVM sparse IR) requires integration with model compression tools
Edge Impulse EON [49]	8-bit Quantization (automatically generated)	Relies on EON Tuner for architecture search optimization and does not directly support pruning
Arm Ethos-U [50]	INT8 Quantization supports mixed precision	Structured pruning (requires preprocessing tools), implemented via the Vela compiler for weight clustering and compression

Table 3. Compression Methods and Toolchain Support.

Compression Method	Toolchain Support
Quantify	TensorFlow Lite [51] (PTQ/QAT) ONNX Runtime [52] (INT8) Intel OpenVINO [53] NVIDIA TensorRT [54] (INT8/FP16)
Structured Pruning	TensorFlow Lite (TF-MOT) NVIDIA TensorRT Arm Ethos-U
Unstructured pruning	TVM ONNX Runtime
Mixed Precision Optimization	NVIDIA TensorRT Arm Ethos-U
Model Reconstruction	LiteRT TVM

In changing the computing method, event-driven algorithms have emerged as an effective optimization approach. By processing only the changed portions of input data, they avoid redundant computations, significantly reducing energy consumption and response latency. Guo et al. [55] proposed a low-power anomaly detection method based on autonomous data fusion from motion sensors, which only wakes up the MCU when abnormal vibrations are detected, ensuring real-time response while reducing system power consumption. Dere et al. [56] describe an event-driven deep neural network that can be deployed on field-programmable gate arrays to classify motion intentions obtained from biosensors. Additionally, distributed federated learning, as an alternative to centralized machine learning, is highly suitable for edge AI. In multi-device systems, module-level sleep strategies can be implemented by dividing power domains based on functionality, enabling sensors or peripherals to be completely powered down when not in use. Alternatively, controllers can be awakened via software interrupts to reactivate corresponding modules

upon event triggering. By modeling multimodal data from different edge devices, data barriers between devices are broken down. This alleviates communication burdens while enhancing data privacy protection capabilities, representing an important direction for the development of lightweight intelligent systems. Huang et al. [57] noted that federated learning can jointly utilize multimodal data from different clients for modeling, improving machine learning efficiency and privacy. Adhikary et al. [58] conducted memory-enhanced device-side federated training on the gray wolf optimizer for embedded intelligent models, achieving an average accuracy improvement of 10.8%. These algorithmic innovations ensure that AI models can still function under the limited resource constraints of MCUs and lay the foundation for future hardware optimizations.

2.2. AI Acceleration Hardware and Heterogeneous Architectures

To resolve the structural contradiction between computational power enhancement and power consumption control, hardware acceleration architectures are evolving toward heterogeneous computing architectures that integrate specialized accelerators and new storage technologies, thereby achieving a significant improvement in system efficiency [59].

When introducing AI into MCUs, the choice between standalone AI modules and integrated AI architectures significantly impacts power consumption and system design. Standalone AI accelerators (such as external NPUs or neuromorphic processors) typically feature extremely low always-on power consumption, achieving active power consumption below 1 mW in event-driven scenarios [60]. This makes them suitable for ultra-low-power devices with stringent battery life requirements. However, this approach introduces additional communication latency and interface power consumption due to data exchange with the MCU via buses like SPI, PCIe, or DMA, while also increasing system integration complexity. In contrast, integrated AI architectures embed acceleration units like NPUs or DSPs directly within the MCU. This allows sharing of on-chip SRAM and bus resources with the MCU core, enabling low-latency, high-bandwidth data access. Through multi-power-domain management, the CPU remains dormant during inference while the NPU handles approximately 95% of computational tasks, significantly reducing data movement energy consumption and system-level power. For instance, the Renesas Cortex-M85 + Ethos-U55 MCU achieves approximately 35 times higher AI inference throughput than CPU-only solutions. A prototype system based on Cortex-M55 + Ethos-U55 even demonstrates about 400 times lower single-inference energy consumption [61]. Therefore, in low-power MCU scenarios, integrated AI architectures are more suitable for applications demanding high real-time performance and energy efficiency, while standalone AI modules are better suited for event-driven scenarios or those with extreme power constraints. The choice of specific solutions should comprehensively consider energy efficiency, latency, system complexity, and application requirements.

Heterogeneous computing architectures can leverage the collaborative effects of different core modules. For example, by coordinating the collaboration of multiple cores such as CPUs, LUTs, and DSPs, dynamic allocation of computational power and task optimization can be achieved, thereby balancing performance and power consumption. Consumer-grade wearables (smartwatches, smart rings, etc.) typically employ ultra-low-power MCUs/SoCs, with a focus on optimizing static standby and sleep energy consumption. For example, the Ambiq Apollo MCU (Cortex-M4F) used in Fossil smartwatches consumes only about 34 μA/MHz during active operation, with sleep current as low as 140 nA. enabling exceptionally long standby lifetimes for wearables during daily use. In contrast, AI acceleration hardware often integrates parallel computing units: RISC-V multi-core platforms incorporate multi-core DSP extensions that significantly boost computational throughput in parallel modes; CIM chips fuse memory and computation, performing matrix operations

via crossbar memory arrays. Some cutting-edge CIM designs achieve energy efficiency at the thousand TOPS/W level within compact footprints. These designs suit scenarios requiring high-density AI inference, but typically exhibit higher leakage current during idle periods and higher current peaks during active periods compared to wearable chips specifically optimized for standby. Ali Farahani et al. [62] introduced a CNNX hardware accelerator optimized for deep separable convolutions, utilizing 8-bit integer Quantization and block processing to address resource constraints and the balance between performance and resource consumption in edge device CNN inference. Zhang et al. [63] developed a new resource management framework, HeteroEdge, to address heterogeneity by effectively allocating tasks to heterogeneous edge devices. Meanwhile, the RISC-V open-source instruction set architecture, with its high customizability and lightweight features, has become the mainstream platform choice for edge AI devices [64]. Yang [65] used RISC-V cores to reduce memory access time through appropriate memory partitioning and allocation. El Zarif et al. [66] introduced Polara-Keras2c, which integrates RISC-V vector extension optimizations and is customized for the Polara architecture, becoming a transformative tool for real-time, energy-efficient AI processing on edge devices and driving the development of edge computing.

To address the low-power requirements of MCUs, dedicated NPUs are often designed with a compute-in-memory architecture, reducing data movement overhead between memory and compute units at the architectural level, thereby significantly lowering energy consumption. Manor Erez et al. [67] proposed a framework combining MCUs and NPUs to resolve the trade-off between resource constraints and performance requirements for neural network deployment on edge devices, achieving up to a 724-fold improvement in inference speed compared to pure software implementations. Lee et al. [68] proposed a convolutional neural network-based super-resolution accelerator for real-time upscaling to ultra-high-definition resolution on edge devices, supporting up to 96 frames per second at UHD resolution. In addition to compute-in-memory architectures, Meng et al. [69] developed a biomimetic "electronic whisker" system that mimics the whiskers of rodents, as shown in Figure 4a, integrating a new type of sensor that combines sensing, storage, and computing. This system can precisely perceive changes in environmental parameters without external power supply. This self-powered intelligent sensor achieves synchronous perception and encoding of multi-modal environmental signals, not only reducing the energy and computational dependencies of traditional sensing systems but also providing a reliable new method for medical monitoring. For example, attaching electronic whiskers to a patient's skin can be used for real-time monitoring of physiological indicators such as respiratory airflow and skin moisture, with potential applications in vital sign monitoring and rehabilitation assessment. This sensing-storage-computing integrated architecture, combined with the storage-computing integrated architecture, further reduces energy consumption.

"MCU + AI" accelerates the intelligent upgrade of automated equipment, achieving higher production efficiency and lower operational costs. Predictive maintenance is one of the most representative applications. Traditional factory equipment maintenance is often conducted on a scheduled or post-failure basis, which can lead to issues such as failure to detect faults in a timely manner or excessive maintenance. Now, by deploying MCUs and embedded AI models on equipment, it is possible to analyze real-time data such as vibration and current of critical components like motors and electric pumps, enabling early prediction of fault trends. Texas Instruments recently launched TMS320F28P55x series MCUs are the first to integrate an NPU, enabling efficient local execution of deep learning inference [70], as shown in Figure 4b. Thanks to NPU hardware acceleration, fault detection latency is significantly reduced, enabling completion within milliseconds—a 5–10× speedup compared to pure software implementations. This low-latency, high-precision local AI inference is

highly suitable for real-time monitoring requirements in industrial environments. These examples demonstrate that hardware architecture innovations combining MCUs with AI are breaking through the constraints of computational power and power consumption, providing efficient support for edge intelligence across various fields.

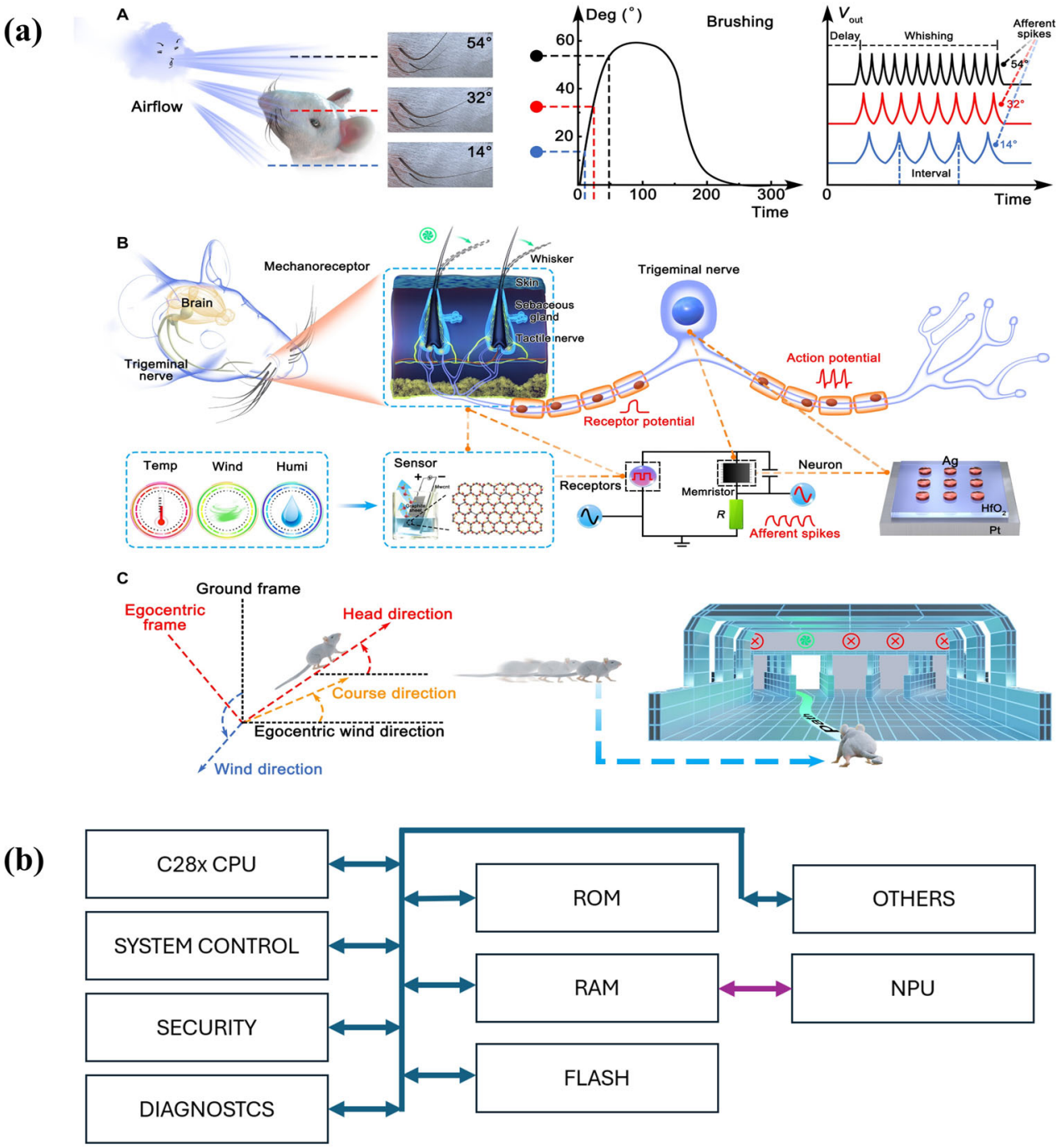


Figure 4. (**a**) Meng et al. invented the schematic illustration of the rat whiskers for wind sensation [69]; (**b**) Functional Block Diagram of TMS320F28P559SJ-Q1. Adapted from Figure 3-1 in [70]; modified to highlight the NPU module.

Through hardware and software co-design, deploying AI in MCUs can effectively compensate for insufficient computing power and control power consumption within acceptable limits by making full use of lightweight models and hardware acceleration, providing a practical solution for edge intelligence. On this basis, the next step will

be to explore at the system level how AI can further empower MCUs to achieve low-power optimization.

3. AI-Enabled Low-Power Optimization Path for MCUs

Leveraging AI technology to optimize the overall power consumption of MCUs is a key approach to achieving low-power operation of devices. With the deep integration of AI models and embedded systems, intelligent algorithms can be used to coordinate task scheduling and power management, significantly improving the system's energy efficiency ratio. This chapter focuses on "How AI Helps MCUs Achieve Low Power Consumption," exploring optimization strategies in areas such as task scheduling, power management, inference engines, communication, and data processing.

3.1. Intelligent Task Scheduling

With the integration of AI models and embedded systems, the deep synergy between AI-driven task scheduling and power management has become a key path to optimizing system energy efficiency. This process essentially relies on operating system scheduling, optimized battery management strategies, and inference engine technology to enhance the real-time performance, efficiency, and accuracy of the overall system.

Optimizing task scheduling at the operating system level ensures the system's responsiveness. Real-time operating systems must support context-aware dynamic task priority adjustment mechanisms to meet stringent real-time response requirements. In scenarios involving concurrent multi-tasking and multi-device collaboration, operating systems must also possess distributed task scheduling capabilities, particularly in multi-core embedded systems where multiple processor cores exist. The real-time scheduling algorithms within these systems are critical for performance and functionality. Xu et al. [71] optimized the scheduling efficiency of multiple tasks across multiple processor cores through algorithmic improvements, fully utilizing multiple processor cores, but the reduction in average time remains limited. Zhu et al. [72] proposed a structure-aware task scheduling strategy for heterogeneous distributed embedded systems, achieving a reduction in task scheduling time of approximately 40% compared to Xu's approach. Zhang et al. [73] proposed a target-sorting-based particle swarm optimization algorithm (TS-MOPSO) and designed a task scheduling model to optimize task scheduling and execution time. Compared with other algorithms, the task execution time, maximum completion time, and total task scheduling cost were reduced by 31.6%, 23.1%, and 16.6%, respectively. These intelligent scheduling strategies effectively enhance the resource utilization and scheduling efficiency of multi-core MCUs, meeting real-time requirements while avoiding energy waste caused by unnecessary idle cycles of processing units.

3.2. Intelligent Power Management

The intelligentization of power management strategies is of great significance for enhancing the overall battery life of a system. Traditional DVFS mechanisms [74] can be further evolved into mechanisms that adapt to changes in load, achieving feedforward control of system computing power requirements by introducing lightweight AI learning models. This enables proactive adjustment of operating voltage and frequency before load changes occur, thereby optimizing the balance between performance and power consumption [75]. Huang et al. [76] proposed a dual-Q power management method that extends the operating frequency based on learning, achieving a total energy savings of 5–18%. Additionally, DVFS methods that further adopt deep reinforcement learning exhibit better performance and robustness. Chen et al. [77] proposed a deep reinforcement learning-based dynamic voltage and frequency scaling method that optimizes the application execution quality

of energy harvesting edge devices, reducing runtime by 17.9% and improving quality by 22.05%. These intelligent power management strategies effectively reduce MCU energy consumption under dynamic loads and enhance energy utilization efficiency.

In addition to chip-level DVFS, adaptive sleep–wake strategies can also be optimized at the system level using AI. By predicting task idle periods through machine learning, the MCU can proactively enter a low-power sleep state and quickly wake up when needed, thereby reducing unnecessary power consumption. AI-assisted control strategies effectively optimize the energy utilization efficiency of devices. At a broader control level, AI can also help optimize power consumption at the device or system level. For example, in the smart home field, the introduction of machine learning for intelligent control of electrical appliances has achieved significant energy-saving effects.

3.3. Inference Engine Optimization

To efficiently run AI inference tasks on an MCU, software-level optimizations are also required. Optimizing the inference engine can also be highly beneficial for embedded platforms with limited computational power. By merging data processing and machine learning tasks using operator fusion technology, the execution efficiency of the inference engine can be significantly improved. Chen et al. [78] proposed the TVM compiler, which optimizes at both the graph level and operator level, providing performance portability for deep learning workloads across different hardware backends. Sun et al. [79] further proposed a GPU-accelerated operator fusion method for evaluating relational query linear algebra, leveraging linear algebra computational properties to merge operators in machine learning predictions and data processing, thereby significantly increasing the runtime speed of the prediction pipeline by 317 times. Similarly, we can also apply the operator fusion method to the MCU platform to optimize it at the software level. On resource-constrained platforms such as MCUs, pre-compiling and optimizing model operators offline can also greatly reduce the instruction overhead and memory accesses during real-time inference. For example, optimization libraries for ARM Cortex-M (such as CMSIS-NN) manually optimize operators like convolution to multiply inference performance [47]. Selecting neural network operators suitable for the MCU architecture, leveraging compiler-automated tensor optimization, and reusing operator computation results as much as possible are all effective ways to enhance inference engine efficiency and reduce energy consumption.

3.4. Communication and Data Compression Optimization

MCUs in edge devices often need to process and transmit large amounts of sensor data. Therefore, reducing transmission volume and lowering communication energy consumption from the data source is also an important part of AI-enabled low power consumption. In video scenes, the inter-frame redundancy rate often exceeds 50%, particularly when the camera is stationary or background changes occur slowly. By transmitting only key frames or regions of interest, data volume can be reduced by several orders of magnitude [80]. In health monitoring, physiological signals such as ECG and body temperature typically exhibit periodic fluctuations with gradual short-term changes. Simple algorithms can compress ECG signals to reduce data volume by approximately 90% while preserving complete heart rhythm characteristics [81]. Environmental sensors collecting temperature, humidity, and light intensity often exhibit long-term correlations. This redundancy can be eliminated through methods like prediction, differential coding, and sparsification. A series of data preprocessing and compression technologies are used to minimize data volume while ensuring that critical information is not lost. First, signal filtering can remove noise interference from the original collected data, improving data quality and avoiding the use of bandwidth and processing power by invalid data. For example, in industrial IoT

applications, noise suppression of sensor signals can significantly enhance the accuracy of subsequent fault analysis [82]. Second, feature dimension reduction compresses high-dimensional data by extracting primary information components, drastically reducing storage and transmission requirements while preserving useful information as much as possible. Ali et al. [83] demonstrated that the dimension reduction of image data at the edge (e.g., using autoencoders or principal component analysis) can reduce data volume by approximately 77% with minimal impact on cloud-based analysis results. Third, adaptive compression strategies can further enhance efficiency by addressing variations in the importance of data components. For example, surveillance footage can be divided into regions of interest (ROIs) and non-ROIs: lossless or low-loss compression can be applied to critical regions, while high-compression-ratio lossy compression can be used for non-critical regions. Such "region of interest" compression significantly improves the overall compression rate without sacrificing important information, thereby markedly enhancing communication bandwidth utilization efficiency [84]. Through these measures, edge devices only need to transmit refined key information, greatly reducing the data exchange volume in wireless communication. Statistics show that preprocessing and filtering redundant data at the edge can significantly reduce bandwidth occupancy and energy consumption caused by the transmission of invalid data. Therefore, combining AI-based local data processing and compression technologies not only alleviates storage and bandwidth bottlenecks in MCUs but also directly reduces energy consumption caused by communication and data storage. In practical applications, this means devices can operate longer on battery power and reduce power consumption caused by frequent communication, thereby improving the overall energy efficiency of the system. Data optimization for edge intelligence requires balancing dimensionality reduction for efficiency with information retention, which has become an important research direction for edge AI deployment [85].

AI technology supports low-power operation of MCUs at multiple levels: by intelligently scheduling to fully utilize hardware resources and avoid idle waste, optimizing power management through learning-based strategies to dynamically balance performance and power consumption, introducing efficient inference engines and compilation optimizations to accelerate algorithm execution, and reducing communication overhead through local data compression and filtering. These approaches work in tandem to enable "MCU + AI" systems to achieve both intelligent decision-making capabilities and low power consumption. In numerous real-world cases, AI-enabled MCU systems have demonstrated significant improvements in energy efficiency and extended battery life. It is foreseeable that as related technologies continue to mature, the intelligent upgrading of MCUs will achieve the dual goals of low power consumption and high performance in an increasing number of scenarios. Finally, Table 4 summarizes the comparison of AI-based low-power technologies.

Table 4. Summary comparing various AI-based low-power techniques.

Solution	Advantages	Disadvantages	Performance Metrics	Source
RL-based DVFS	Adjust speed and voltage according to real-time demand to save active power consumption	Switching delay and voltage stability; achieving complex	Energy consumption reduced by 5–18%, operating time decreased by 17.9%	[76,77]

Table 4. *Cont.*

Solution	Advantages	Disadvantages	Performance Metrics	Source
Predictive Task Scheduling	Resource utilization has been significantly optimized; Task response delays have been substantially reduced	Additional computational and data overhead; Data privacy and security risks	Task execution time reduced by 31.6%, scheduling time reduced by 40%	[72,73]
Lightweight Model	Reduce computational load and memory usage to extend battery life	Accuracy decreases, requiring more complex preprocessing or model tuning	Model size reduced by a factor of 19, number of parameters reduced by a factor of 13,960	[45,86]
NPU Acceleration	Dedicated MAC array accelerates inference with low latency	Increase chip area and static power consumption; Initialization/switching overhead	Inference speed increased by 724 times	[67]
CIM Architecture	Significantly reduce data transport energy consumption and improve MAC efficiency	New process and analog circuit challenges; affected by PVT variations	Energy efficiency reaches the level of 1000 TOPS per watt	[87]

4. MCU + AI Enables Low-Power Technology Applications

4.1. Deep Application of Smart Home Scenarios

In smart homes, MCUs combined with AI technology have significantly improved security and energy management capabilities. On the one hand, security systems based on visual and behavioral analysis have made home monitoring smarter. Cameras installed in the home environment use edge AI to analyze video streams in real time. They can automatically identify abnormal events (e.g., illegal intrusions, fires) and promptly raise the alarm. AI-driven intrusion detection systems can learn normal household activity patterns to detect abnormal intrusions or suspicious movements [88], as shown in Figure 5a. Additionally, cameras can perform personnel identity verification and behavior analysis to implement more advanced home security strategies, while local processing reduces energy consumption caused by long-term wireless transmission of video streams. On the other hand, AI-assisted control strategies effectively optimize the energy efficiency of devices. A typical example is HVAC systems such as air conditioners: by obtaining status information from environmental sensors and combining it with user preferences, AI models can adjust air conditioner operating parameters in real time, starting, stopping, and adjusting power only when necessary. It has been reported that this intelligent control strategy can reduce the energy consumption of air conditioning systems by approximately 20–30% [89]. Another example is the GD32G5 series MCU launched by GigaDevice, which integrates a high-performance Arm Cortex-M33 core and DSP accelerator and adopts a low-power design, enabling efficient local execution of energy-efficient optimization algorithms [90]. This MCU can quickly calculate the optimal power output based on data from multiple sensors and precisely control the operation of components such as compressors and motors, thereby making home appliances run more efficiently and stably. "MCU + AI" enables smart home devices to more intelligently perceive their environment and user needs, enhancing safety while achieving energy-efficient operation.

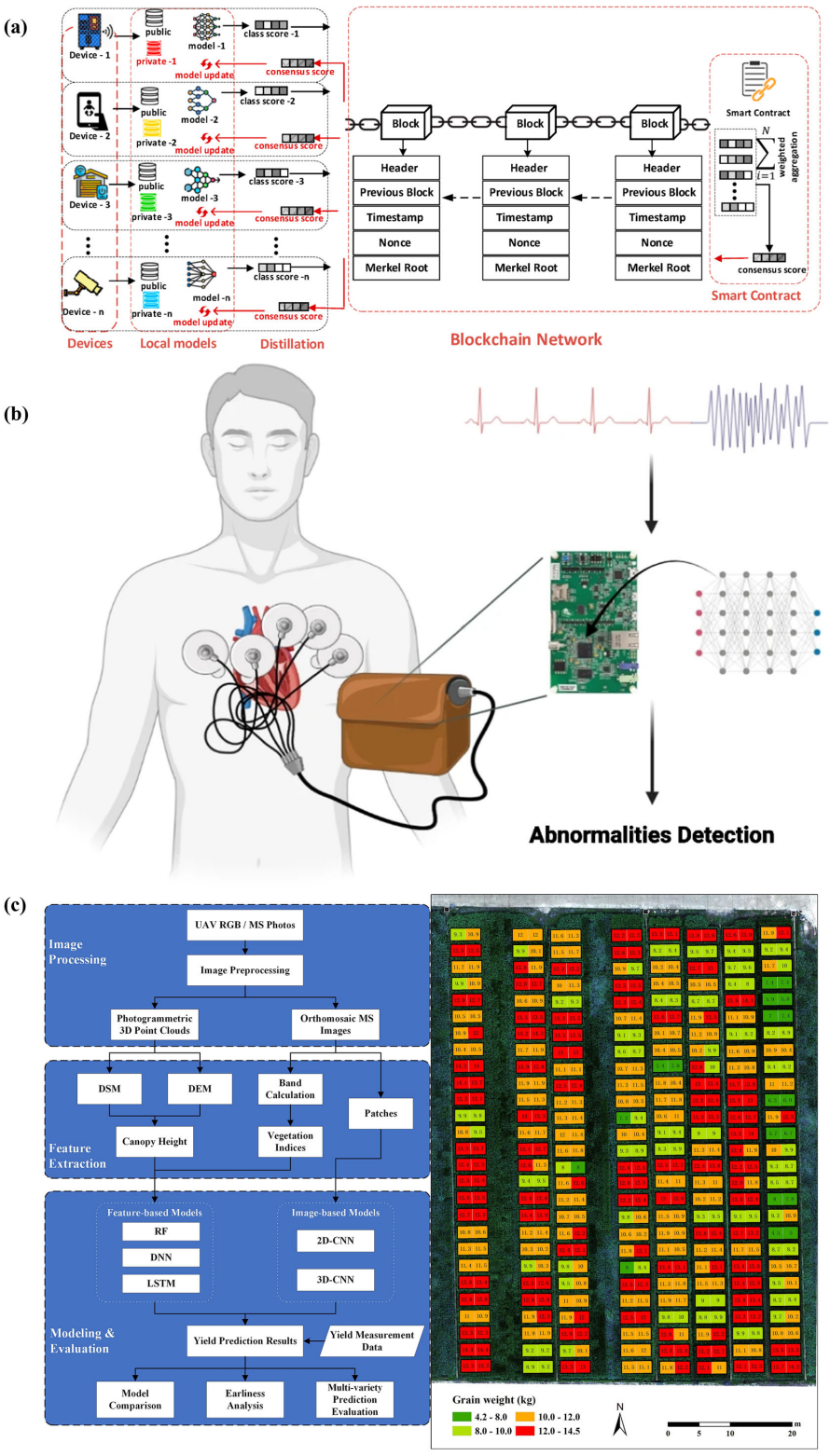


Figure 5. (**a**) Shalan et al. employs a comprehensive system that integrates knowledge distillation, transfer learning, and blockchain technology to address the unique challenges of smart home intrusion detection [88]; (**b**) Huang et al. introduce two models, ConvLSTM2D-liquid time-constant network and ConvLSTM2D-closed-form continuous-time neural network, designed for abnormality identification using electrocardiogram data [91]; (**c**) Flowchart for yield prediction modeling and spatial map of yield prediction based on the CNN-M2D model. Each plot was divided into two panels, with the left side displaying the observed yield data, and the right side showing the predicted yield data. Reproduced with permission [92]. Copyright 2024 Elsevier.

4.2. Intelligent Innovation in Medical Devices

The integration of MCUs and AI technology is driving the development of medical devices toward greater intelligence and precision. In the field of physiological signal monitoring, wearable medical devices extensively utilize embedded AI algorithms to enable early disease warning, as shown in Figure 5b. In battery-powered wearable medical devices, MCUs can continuously run embedded AI models to perform local analysis of electrocardiogram (ECG) signals for detecting arrhythmias such as atrial fibrillation [93]. Compared to traditional Holter monitoring, which requires post hoc manual analysis, this significantly improves the detection rate and timeliness of abnormal heart rhythms. Antoun et al. [94] demonstrated that analyzing wearable ECG data using AI can significantly enhance the early detection rate of asymptomatic arrhythmias such as atrial fibrillation. AI algorithms can identify subtle abnormal patterns from massive amounts of dynamic ECG data, enabling early warning of severe heart disease. On the other hand, implantable medical devices have also incorporated intelligent control. For example, the next-generation artificial pancreas combines an MCU and AI algorithms to automatically adjust insulin release based on real-time physiological data, achieving closed-loop blood glucose control [95]. Similarly, research has explored the use of AI in implantable devices such as pacemakers to adaptively adjust stimulation parameters or drug release strategies based on patient status, thereby improving therapeutic efficacy. "MCU + AI" is driving the transformation of medical devices toward miniaturization and intelligence, with various devices demonstrating higher autonomy and accuracy from wearable monitoring to implantable therapy, while meeting low-power requirements and maintaining extended operational durations.

4.3. Digital Transformation of Smart Agriculture and Industry

In the field of smart agriculture, AI technology optimizes irrigation and fertilization strategies through environmental monitoring (temperature, humidity, soil moisture), and combines computer vision to enable early detection of crop pests and diseases. For example, high-definition cameras or drones installed in fields regularly capture images of crops, while microcontrollers run visual models such as convolutional neural networks in real time at the edge to automatically identify disease spots or pest traces on crop leaves. Venkateswara et al. [96] demonstrated that deep learning models can achieve high accuracy in crop pest and disease image recognition, enabling timely detection of early infection signs. Compared to manual inspections, this approach can identify issues several days earlier, significantly reducing pest and disease-related losses. Additionally, integrated air-ground-space monitoring makes crop growth assessment and yield prediction more scientific. Drones controlled by microcontrollers equipped with multispectral cameras conduct regular aerial surveys of farmland, capturing panoramic growth parameters of crop canopies, such as the vegetation index NDVI. Edge devices utilize machine learning models to fuse multi-temporal remote sensing data with ground sensor data, establishing a relationship model between crop growth and yield. Zhou et al. [92] demonstrated that a method combining drone imagery and deep learning can accurately predict yield per unit area, as shown in Figure 5c. By integrating drone multispectral images with deep neural networks, they achieved yield predictions for rice with over 95% accuracy. This technology enables agricultural departments and farmers to anticipate yield information in advance, optimizing harvest and sales plans. The "microcontroller + AI" smart agriculture solution enables comprehensive perception of agricultural conditions and intelligent decision-making: it not only "manages fields based on weather conditions" by optimizing water use and fertilization through environmental data, but also "detects diseases and pests" by using visual intelligence to prevent and control pests and diseases early,

thereby promoting agricultural productivity and efficiency and driving the transformation of traditional agriculture toward digitalization and precision agriculture.

In the industrial and energy sectors, AI-enabled control strategies also play a role in reducing energy consumption. In photovoltaic power generation systems, intelligent algorithms running on MCUs can adjust the operating point of photovoltaic panels at high speed to track maximum power output. Li et al. [97] combined maximum power point tracking (MPPT) algorithms with intelligent control strategies, significantly improving the efficiency and reliability of solar power systems and extending battery lifespan. As such, whether in power management at the chip level or energy optimization at the device application level, AI technology offers new approaches and tools for reducing MCU system power consumption.

5. Challenges and Prospects

Deploying AI capabilities on MCUs presents a series of technical challenges that require targeted solutions. The primary challenge is balancing the lightweight nature of AI models with their accuracy and robustness. Large-scale deep learning models typically have a massive number of parameters and complex computations, making them difficult to directly port to MCUs with extremely limited memory and computational power. To adapt to MCUs, model compression techniques (such as Quantization, pruning, and distillation) are needed to reduce the model size to the KB level. However, excessive compression can lead to a significant decline in model accuracy and robustness. Ortiz et al. [86] demonstrate that most model compression methods, if not finely tuned, inevitably sacrifice accuracy. Pruning and Quantization can reduce model size by an average of 19 times, but accuracy may significantly decline. Therefore, the bottleneck lies in how to minimize model complexity while maintaining sufficient accuracy. On one hand, more advanced model compression and neural architecture search techniques need to be developed to ensure that models still perform reliably in resource-constrained environments. On the other hand, to address edge noise and domain shift phenomena, robustness needs to be enhanced during the model training phase to prevent lightweight models from being overly sensitive to environmental changes. In summary, achieving a balance between “lightweight” and “high accuracy” is one of the core challenges of MCU intelligence.

Second, consistency and stability issues in edge-side distributed collaborative computing have become prominent. Multi-cluster communication refers to the interconnection and collaboration of multiple MCUs. This architecture allows each node to enter sleep mode independently, offering high flexibility but introducing communication latency and protocol overhead. Performance gains from collaboration must account for communication synchronization overhead. For example, when multiple nodes process different tasks in parallel, computational power scales linearly, but this incurs additional transceiver power consumption and communication latency costs. System-level simulation or actual testing can measure total power consumption and task completion time under scenarios involving single-node, single-cluster (no communication), and multi-cluster communication. This allows comparison of how communication latency and energy consumption impact system efficiency. When multiple MCU nodes collaborate to execute AI tasks to balance the load, network latency and different node states may cause data inconsistency and timing asynchrony, affecting overall inference accuracy and system stability [98]. To overcome this bottleneck, it is necessary to study high-fault-tolerant, low-latency edge collaboration mechanisms, including efficient parameter synchronization algorithms and consistency maintenance protocols. Recent work has made progress in end-to-edge collaborative inference by dynamically dividing models for joint inference across devices and edge servers to reduce single-node load and response latency. However, even under this architecture,

performance fluctuations caused by dynamic factors such as network jitter and node failures must still be addressed. Future research directions include the following: designing adaptive collaborative scheduling strategies to dynamically adjust task allocation based on network conditions; introducing distributed consensus and state verification mechanisms to ensure consistency in processing and reliability of results across nodes. Additionally, for multi-node deployments, data consistency and system security must be considered. Only by addressing stability and consistency challenges in edge collaborative computing can MCU clusters fully leverage economies of scale to achieve reliable distributed intelligence.

Finally, reliability and power consumption control in the "sensing, storage, and computing integrated" architecture remain areas requiring breakthroughs. The deep integration of sensing, storage, and computing units is widely regarded as a core path to enhancing energy efficiency in next-generation intelligent systems [99]. Among various implementation schemes for this integrated architecture, in-memory computing (CIM) stands out for its potential to fundamentally reduce data movement between discrete units—by embedding computational operations directly within storage arrays, it can significantly lower overall energy consumption and improve computational throughput. Specifically, the use of novel non-volatile memory devices (such as memristors) to construct crossbar arrays enables critical operations like vector-matrix multiplication to be performed within the storage array, realizing a "near-storage-as-computing" paradigm that aligns with the energy-saving goals of the integrated architecture. However, this CIM-based implementation of the sensing-storage-computing integrated architecture currently faces severe challenges in reliability and power consumption control—and these challenges are exactly the core bottlenecks that the integrated architecture itself needs to overcome. In the sensing-storage-computing integrated architecture, the accuracy of sensing signals, the stability of stored data, and the correctness of computational results are highly coupled—for example, drift in memristor conductance will not only distort stored information but also propagate errors to the computing process, while noisy sensing signals will further exacerbate these uncertainties. These random fluctuations and drift phenomena introduce unavoidable errors in analog computing, and addressing and tolerating such uncertainties at the circuit and algorithm levels remains a major challenge for the entire integrated architecture. Ding et al. [100] pointed out that the randomness of ion migration processes within memristors can cause inconsistencies in device switching thresholds and delays—a problem that is amplified in large-scale CIM systems (the core of the integrated architecture), severely affecting the stable coordination of sensing, storage, and computing units. To enhance the reliability of the integrated architecture, the academic community is exploring variation-tolerant and error-correction techniques, such as introducing redundant storage units, error detection, and erasure coding within CIM arrays (to compensate for device variability), or calibrating memristors through multiple programming and read/write cycles (to mitigate drift effects)—all of which are targeted at resolving reliability bottlenecks of the integrated system. In addition to reliability, power consumption control is another critical challenge for the sensing-storage-computing integrated architecture. Although the energy consumption of a single CIM array is very low, the integrated architecture requires the parallel and collaborative operation of sensing modules, multiple CIM arrays, and auxiliary circuits. When a large number of CIM arrays work in parallel to process continuous signals from sensing units, the cumulative power consumption of the entire system remains significant. Moreover, analog circuits in the CIM array consume additional energy during standby and refresh processes—this "idle energy consumption" is particularly problematic for low-power scenarios that the integrated architecture aims to serve. Developing precise power management strategies is therefore essential: these strategies need to dynamically allocate power across sensing, storage, and computing units (e.g., adjusting the operating

frequency of CIM arrays based on sensing data throughput, or putting idle sensing modules into low-power mode), ensuring that the integrated system maintains high throughput without causing local overheating or uncontrolled energy consumption. Crafton et al. [87] further noted that the CIM-based integrated architecture introduces new power-related issues not encountered in traditional CMOS systems—for instance, computational errors caused by device variation require additional retries, which actually increase the average power consumption per effective computation. This means that while pursuing extreme energy efficiency, it is necessary to incorporate power-aware scheduling and calibration mechanisms into the system design—balancing reliability and low power consumption across the entire sensing-storage-computing chain.

For resource-constrained embedded scenarios, it is necessary to comprehensively consider task characteristics, hardware resources, and accuracy requirements to select or customize suitable AI algorithm architectures. Model compression can significantly reduce power consumption by lowering model memory usage and computational load, but this often comes at the cost of reduced accuracy, requiring careful balancing between model size and accuracy [101]. In recent years, the industry has also proposed methods for algorithm-hardware co-design, such as the MCUNet framework, which combines neural architecture search and memory optimization to enable commercial microcontrollers to successfully run ImageNet-level models with over 70% Top-1 accuracy, while reducing SRAM and Flash memory usage by 3.5 times and 5.7 times, respectively [102]. This demonstrates that through lightweight model design optimized for MCU constraints, high intelligent performance can be achieved on limited resources.

Secondly, the balanced application of hardware acceleration solutions. To enhance the intelligent processing capabilities of MCUs, heterogeneous multi-core architectures or dedicated accelerators can be introduced, but it is important to balance performance improvements with cost and power consumption. Heterogeneous architectures can significantly improve hardware resource utilization and reduce power consumption by enabling different cores to collaborate and divide tasks [103]. For example, a study proposed an NPU-MCU collaborative computing framework that leverages task partitioning and pipelining optimizations to fully utilize NPU computing power and MCU control capabilities, significantly enhancing system computing efficiency and energy efficiency. In practical applications, appropriate hardware acceleration methods should be selected based on application requirements to balance performance improvements with power consumption and cost constraints.

Finally, there is intelligent tuning of DVFS. Given the time-varying characteristics of AI workloads, establishing an accurate power consumption prediction model and combining it with real-time task requirements to dynamically adjust MCU voltage and frequency can maximize energy efficiency. This strategy requires software-hardware collaboration: on the one hand, monitoring task load changes and promptly reducing frequency and voltage to save energy; on the other hand, enhancing performance during critical computation phases to ensure real-time responsiveness. According to the latest research, incorporating predictive models or reinforcement learning into DVFS decision-making can reduce energy consumption by approximately 20% while maintaining performance metrics, with minimal impact on real-time response and a frame drop rate controlled below 2% [104]. Therefore, DVFS optimization for MCU intelligence is a key strategy for enhancing overall energy efficiency, requiring careful consideration of power consumption model accuracy and response speed to achieve the optimal dynamic balance between energy consumption and performance.

Author Contributions: Conceptualization, X.L. and T.Z.; methodology, Y.W. and T.Z.; validation, X.L. and T.Z.; formal analysis, T.Z.; investigation, Y.W. and T.Z.; resources, B.H. and T.Z.; data curation,

J.L. and J.F.; writing—original draft preparation, T.Z.; writing—review and editing, Y.W. and T.Z.; supervision, T.Z. and Z.Y.; project administration, Y.W. and Z.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new data were created or analyzed in this study.

Acknowledgments: We would like to thank Zhetong Cao for his assistance during his time in the laboratory, particularly for providing general support and participating in learning activities.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhang, H.Z.; He, M.X.; Shi, L.L.; Wang, P.F. Terahertz Thickness Measurement Based on Stochastic Optimization Algorithm. *Spectrosc. Spectr. Anal.* **2020**, *40*, 3066–3070. [CrossRef]
2. Silva, S.N.; Goldbarg, M.A.S.d.S.; da Silva, L.M.D.; Fernandes, M.A.C. Real-Time Simulator for Dynamic Systems on FPGA. *Electronics* **2024**, *13*, 4056. [CrossRef]
3. Krejci, J.; Babiuch, M.; Babjak, J.; Suder, J.; Wierbica, R. Implementation of an Embedded System into the Internet of Robotic Things. *Micromachines* **2023**, *14*, 113. [CrossRef]
4. Cheshfar, M.; Maghami, M.H.; Amiri, P.; Garakani, H.G.; Lavagno, L. Comparative Survey of Embedded System Implementations of Convolutional Neural Networks in Autonomous Cars Applications. *IEEE Access* **2024**, *12*, 182410–182437. [CrossRef]
5. Son, J.H.; Cho, Y.H.; Kim, J.H.; Joo, Y.S.; So, S.S. Design and Implementation of a sensor node for out-door environmental monitoring. *Trans. Korean Inst. Electr. Eng. P* **2007**, *56*, 117–122.
6. Yang, H.; Yao, J.; Liu, J. A novel speech recognition system-on-chip. In Proceedings of the International Conference on Audio, Language and Image Processing, Shanghai, China, 7–9 July 2008; pp. 764–768.
7. Xie, G.; Chen, Y.; Xiao, X.; Xu, C.; Li, R.; Li, K. Energy-Efficient Fault-Tolerant Scheduling of Reliable Parallel Applications on Heterogeneous Distributed Embedded Systems. *IEEE Trans. Sustain. Comput.* **2018**, *3*, 167–181. [CrossRef]
8. Tiwari, A.; Ballal, P.; Lewis, F.L. Energy-efficient wireless sensor network design and implementation for condition-based maintenance. *ACM Trans. Sens. Netw. (TOSN)* **2007**, *3*, 1-es. [CrossRef]
9. Kono, T.; Taito, Y.; Hidaka, H. Essential Roles, Challenges and Development of Embedded MCU Micro-Systems to Innovate Edge Computing for the IoT/AI Age. *IEICE Trans. Electron.* **2020**, *E103C*, 132–143. [CrossRef]
10. Zangi, U.; Feldman, N.; Hadas, T.; Dayag, N.; Shor, J.; Fish, A. 0.45 v and 18 µA/MHz MCU SOC with Advanced Adaptive Dynamic Voltage Control (ADVC). *J. Low Power Electron. Appl.* **2018**, *8*, 14. [CrossRef]
11. Lee, H.; Kim, E.; Lee, Y.; Kim, H.; Lee, J.; Kim, M.; Yoo, H.-J.; Yoo, S. Toward all-day wearable health monitoring: An ultralow-power, reflective organic pulse oximetry sensing patch. *Sci. Adv.* **2018**, *4*, eaas9530. [CrossRef]
12. Flamand, E.; Rossi, D.; Conti, F.; Loi, I.; Pullini, A.; Rotenberg, F.; Benini, L. GAP-8: A RISC-V SoC for AI at the Edge of the IoT. In Proceedings of the 29th Annual IEEE International Conference on Application-Specific Systems, Architectures and Processors (ASAP), Milan, Italy, 10–12 July 2018; pp. 69–72.
13. Zhang, Y.; Yin, B.; Gomony, M.D.; Corporaal, H.; Trinitis, C.; Corradi, F. Hardware/Software Co-Design Optimization for Training Recurrent Neural Networks at the Edge. *J. Low Power Electron. Appl.* **2025**, *15*, 15. [CrossRef]
14. Sharifi, S.; Coskun, A.K.; Rosing, T.S. Hybrid dynamic energy and thermal management in heterogeneous embedded multiprocessor SoCs. In Proceedings of the 2010 15th Asia and South Pacific Design Automation Conference (ASP-DAC), Taipei, Taiwan, 18–21 January 2010; pp. 873–878.
15. Zhao, Z.; Xiao, Z.; Tao, J. MSDG: Multi-Scale Dynamic Graph Neural Network for Industrial Time Series Anomaly Detection. *Sensors* **2024**, *24*, 7218. [CrossRef]
16. Sofianidis, I.; Konstantakos, V.; Nikolaidis, S. Energy Consumption Aspects on Embedded System for IoT Applications. In Proceedings of the 13th International Conference on Modern Circuits and Systems Technologies (MOCAST), Sofia, Bulgaria, 26–28 June 2024.
17. Capogrosso, L.; Cunico, F.; Cheng, D.S.; Fummi, F.; Cristani, M. A Machine Learning-Oriented Survey on Tiny Machine Learning. *IEEE Access* **2024**, *12*, 23406–23426. [CrossRef]
18. Jiang, B.-Y.; Zhou, F.-C.; Chai, Y. Application of neuromorphic resistive random access memory in image processing. *Acta Phys. Sin.* **2022**, *71*, 148504. [CrossRef]
19. Gebregiorgis, A.; Hoang Anh Du, N.; Yu, J.; Bishnoi, R.; Taouil, M.; Catthoor, F.; Hamdioui, S. A Survey on Memory-centric Computer Architectures. *ACM J. Emerg. Technol. Comput. Syst.* **2022**, *18*, 79. [CrossRef]
20. Li, Z.; Yang, B.; Li, Y. Development of an implantable animal activity rhythm detector and the low-power design. *Sheng Wu Yi Xue Gong Cheng Xue Za Zhi J. Biomed. Eng. Shengwu Yixue Gongchengxue Zazhi* **2011**, *28*, 1121–1125.

21. Parvati, S.V.; Sathish, S.; Thenmozhi, K.; Amirtharajan, R.; Praveenkumar, P. IoT Accelerated Wi-Fi Bot controlled via Node MCU. In Proceedings of the 8th International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 4–6 January 2018.
22. Araujo, D.; Marques, C.; Oliveira, H.; Piteira, J. A flexible and low-power IoT controller for agri-food field sensing applications. In Proceedings of the 2024 Smart Systems Integration Conference and Exhibition, Hamburg, Germany, 16–18 April 2024.
23. Ayers, H.R. Making Small Embedded Systems Secure and Dependable. Ph.D. Thesis, Stanford University, Stanford, CA, USA, 2023.
24. Gomes, P.M.; Brito, M.A. Low-Code Development Platforms: A Descriptive Study. In Proceedings of the 17th Iberian Conference on Information Systems and Technologies (CISTI), Madrid, Spain, 22–25 June 2022.
25. Jung, V.J.-B.; Burrello, A.; Scherer, M.; Conti, F.; Benini, L. Optimizing the Deployment of Tiny Transformers on Low-Power MCUs. *IEEE Trans. Comput.* **2025**, *74*, 526–541. [CrossRef]
26. Situ, L.; Zhang, C.; Guan, L.; Zuo, Z.; Wang, L.; Li, X.; Liu, P.; Shi, J. Physical Devices-Agnostic Hybrid Fuzzing of IoT Firmware. *IEEE Internet Things J.* **2023**, *10*, 20718–20734. [CrossRef]
27. Vangal, S.; Paul, S.; Hsu, S.; Agarwal, A.; De, V. Near-Threshold Voltage Design Techniques for Heterogenous Manycore System-on-Chips. *J. Low Power Electron. Appl.* **2020**, *10*, 16. [CrossRef]
28. Jisu, K.; Park, D.J. Collaborative Streamlined On-Chip Software Architecture on Heterogenous Multi-Cores for Low-Power Reactive Control in Automotive Embedded Processors. *IEMEK J. Embed. Syst. Appl.* **2022**, *17*, 375–382. [CrossRef]
29. Torres, G.M.; Souza, J.V.L.D.; Aguiar, J.G.F.; Sousa, J.O.d. Performance and Power Consumption of Real-Time Task Scheduling for Embedded Systems Devices. In Proceedings of the 2025 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 11–14 January 2025; pp. 1–6.
30. Zhou, F.; Zhao, L.; Ding, X.; Wang, S. Enhanced DDPG algorithm for latency and energy-efficient task scheduling in MEC systems. *Discov. Internet Things* **2025**, *5*, 40. [CrossRef]
31. Nayak, S.; Patgiri, R.; Waikhom, L.; Ahmed, A. A review on edge analytics: Issues, challenges, opportunities, promises, future directions, and applications. *Digit. Commun. Netw.* **2024**, *10*, 783–804. [CrossRef]
32. Sun, X.; Xu, C.; Li, C. Minimizing Peak Memory Footprint of Inference on IoTs Devices by Efficient Recomputation. In Proceedings of the 19th International Conference on Advanced Intelligent Computing Technology and Applications (ICIC), Zhengzhou, China, 10–13 August 2023; pp. 15–26.
33. Boyle, L.; Moosmann, J.; Baumann, N.; Heo, S.; Magno, M. DSORT-MCU: Detecting Small Objects in Real Time on Microcontroller Units. *IEEE Sens. J.* **2024**, *24*, 40231–40239. [CrossRef]
34. Tang, X.; Shang, D. Design of High-performance while Energy-efficient Microprocessor with Novel Asynchronous Techniques. In Proceedings of the 35th IEEE International Conference on Application-Specific Systems, Architectures and Processors (ASAP), City Univ Hong Kong, Dept Elect Engn, Hong Kong, China, 24–26 July 2024; pp. 247–248.
35. Younesi, A.; Fazli, M.A.; Ejlali, A. A Novel Levy Walk-based Framework for Scheduling Power-intensive Mobile Edge Computing Tasks. *J. Grid Comput.* **2024**, *22*, 69. [CrossRef]
36. Abadade, Y.; Temouden, A.; Bamoumen, H.; Benamar, N.; Chtouki, Y.; Hafid, A.S. A Comprehensive Survey on TinyML. *IEEE Access* **2023**, *11*, 96892–96922. [CrossRef]
37. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge Computing: Vision and Challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [CrossRef]
38. Kim, D.; Roh, C.; Baek, D.; Choi, S.-G. Low-Power Preprocessing System at MCU-Based Application Nodes for Reducing Data Transmission. *Electronics* **2024**, *13*, 2932. [CrossRef]
39. Zhang, J.; Xie, X.; Peng, G.; Liu, L.; Yang, H.; Guo, R.; Cao, J.; Yang, J. A Real-Time and Privacy-Preserving Facial Expression Recognition System Using an AI-Powered Microcontroller. *Electronics* **2024**, *13*, 2791. [CrossRef]
40. Oyeronke, A. Trusted Execution Environments for Edge Devices: Architecture and Attacks. Available online: https://www.researchgate.net/publication/393449260_ (accessed on 6 April 2025).
41. Samanta, R.; Saha, B.; Ghosh, S.K. TinyML-On-The-Fly: Real-Time Low-Power and Low-Cost MCU-Embedded On-Device Computer Vision for Aerial Image Classification. In Proceedings of the IEEE Space, Aerospace and Defence Conference (SPACE), Bangalore, India, 22–23 July 2024; pp. 194–198.
42. Kong, H.; Liu, D.; Huai, S.; Luo, X.; Subramaniam, R.; Makaya, C.; Lin, Q.; Liu, W. Edge Compress: Coupling Multidimensional Model Compression and Dynamic Inference for EdgeAI. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2023**, *42*, 4657–4670. [CrossRef]
43. Song, Z.R.; Fu, B.Q.; Wu, F.Y.; Jiang, Z.M.; Jiang, L.; Jing, N.F.; Liang, X.Y. DRQ: Dynamic Region-based Quantization for Deep Neural Network Acceleration. In Proceedings of the 47th ACM/IEEE Annual International Symposium on Computer Architecture (ISCA), Electr Network, Valencia, Spain, 3 May–3 June 2020; pp. 1010–1021.

44. Liu, Z.H.; Wang, Y.H.; Han, K.; Ma, S.W.; Gao, W. Instance-Aware Dynamic Neural Network Quantization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 12424–12433.
45. Xie, Y.-L.; Lin, X.-R.; Lee, C.-Y.; Lin, C.-W. Design and Implementation of an ARM-Based AI Module for Ectopic Beat Classification Using Custom and Structural Pruned Lightweight CNN. *IEEE Sens. J.* **2024**, *24*, 19834–19844. [CrossRef]
46. Krasteva, V.; Stoyanov, T.; Jekova, I. Implementing Deep Neural Networks on ARM-Based Microcontrollers: Application for Ventricular Fibrillation Detection. *Appl. Sci.* **2025**, *15*, 1965. [CrossRef]
47. Sakr, F.; Bellotti, F.; Berta, R.; De Gloria, A.; Doyle, J. Memory-Efficient CMSIS-NN with Replacement Strategy. In Proceedings of the 8th International Conference on Future Internet of Things and Cloud, Rome, Italy, 23–25 August 2021; pp. 299–303.
48. Liu, C.; Jobst, M.; Jobst, M.; Guo, L.Y.; Shi, X.Y.; Partzsch, J.; Mayr, C.; Assoc Computing, M. Deploying Machine Learning Models to Ahead-of-Time Runtime on Edge Using MicroTVM. In Proceedings of the IEEE/ACM International Workshop on Compilers, Deployment, and Tooling for Edge AI (CODAI), Hamburg, Germany, 21 September 2023; pp. 37–40.
49. Diab, M.S.; Rodriguez-Villegas, E. Performance Evaluation of Embedded Image Classification Models Using Edge Impulse for Application on Medical Images. In Proceedings of the 44th Annual International Conference of the IEEE-Engineering-in-Medicine-and-Biology-Society (EMBC), Glasgow, Scotland, 11–15 July 2022; pp. 2639–2642.
50. Arm Limited. Available online: https://community.arm.com/arm-community-blogs/b/ai-blog/posts/pruning-clustering-arm-ethos-u-npu (accessed on 22 July 2023).
51. Sá, P.; Bessa Loureiro, R.; Lisboa, F.; Peixoto, R.; Nascimento, L.; Bonfim, Y.; Cruz, G.; Ramos, T.; Montes, C.; Pagano, T.; et al. Efficient Deployment of Machine Learning Models on Microcontrollers: A Comparative Study of Quantization and Pruning Strategies. In Proceedings of the IX Simpósio Internacional de Inovação e Tecnologia, Salvador, Brazil, 25–27 October 2023; pp. 181–188.
52. Joshua, C.; Karkala, S.; Krishnapatnam, M.; Aggarwal, A.; Zahir, Z.; Pandhare, V.; Shah, V. Cross-Platform Optimization of ONNX Models for Mobile and Edge Deployment. *Int. J. Wirel. Netw. Broadband Technol.* **2025**. Available online: https://www.researchgate.net/publication/392623112 (accessed on 24 September 2025).
53. Pradana, E.Y.; Aji, S.A.; Abdulrrozaq, M.A.; Alasiry, A.H.; Risnumawan, A.; Pitowarno, E. Optimizing YOLOv8 for Real-Time Performance in Humanoid Soccer Robots with OpenVINO. In Proceedings of the 26th International Electronics Symposium (IES) on Shaping the Future—Society 5.0 and Beyond, Politeknik Elektronika Negeri Surabaya, Surabaya, Indonesia, 6–8 August 2024; pp. 304–309.
54. Shafique, M.A.; Munir, A.; Kong, J. Deep Learning Performance Characterization on GPUs for Various Quantization Frameworks. *AI* **2023**, *4*, 926–948. [CrossRef]
55. Guo, J.; Wang, K.; Sun, J.; Jia, Y. Research and Implementation of Low-Power Anomaly Recognition Method for Intelligent Manhole Covers. *Electronics* **2023**, *12*, 1926. [CrossRef]
56. Dere, M.D.; Jo, J.-H.; Lee, B. Event-Driven Edge Deep Learning Decoder for Real-Time Gesture Classification and Neuro-Inspired Rehabilitation Device Control. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 4011612. [CrossRef]
57. Huang, W.; Wang, D.X.; Ouyang, X.C.; Wan, J.H.; Liu, J.; Li, T.R. Multimodal federated learning: Concept, methods, applications and future directions. *Inf. Fusion* **2024**, *112*, 102576. [CrossRef]
58. Adhikary, S.; Dutta, S. *FedTinyWolf*—A Memory Efficient Federated Embedded Learning Mechanism. *IEEE Embed. Syst. Lett.* **2024**, *16*, 513–516. [CrossRef]
59. Song, J.; Xie, G.; Li, R.; Chen, X. An Efficient Scheduling Algorithm for Energy Consumption Constrained Parallel Applications on Heterogeneous Distributed Systems. In Proceedings of the 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC), Guangzhou, China, 12–15 December 2017; pp. 32–39.
60. Emilio, M.D.P. BrainChip's Akida NPU: Redefining AI Processing with Event-Based Architecture. Available online: https://www.embedded.com/brainchips-akida-npu-redefining-ai-processing-with-event-based-architecture (accessed on 1 October 2024).
61. Corporation, R.E. Renesas Sets New MCU Performance Bar with 1-GHz RA8P1 Devices with AI Acceleration. Available online: https://www.renesas.com/en/about/newsroom/renesas-sets-new-mcu-performance-bar-1-ghz-ra8p1-devices-ai-acceleration?srsltid=AfmBOoo7t4p8f49vQPZ8sh3YrOiwjPHn47OYsyw8jTGXtqDBoRP0UxQA (accessed on 1 July 2025).
62. Farahani, A.; Beithollahi, H.; Fathi, M.; Barangi, R. CNNX: A Low Cost, CNN Accelerator for Embedded System in Vision at Edge. *Arab. J. Sci. Eng.* **2023**, *48*, 1537–1545. [CrossRef]
63. Zhang, D.; Rashid, M.T.; Li, X.K.; Vance, N.; Wang, D. HeteroEdge: Taming The Heterogeneity of Edge Computing System in Social Sensing. In Proceedings of the ACM/IEEE International Conference on Internet of Things Design and Implementation (IoTDI), Montreal, QC, Canada, 15–18 April 2019; pp. 37–48.
64. Garofalo, A.; Tortorella, Y.; Perotti, M.; Valente, L.; Nadalini, A.; Benini, L.; Rossi, D.; Conti, F. DARKSIDE: A Heterogeneous RISC-V Compute Cluster for Extreme-Edge On-Chip DNN Inference and Training. *IEEE Open J. Solid-State Circuits Soc.* **2022**, *2*, 231–243. [CrossRef]

65. Yang, C.H. AI Acceleration with RISC-V for Edge Computing. In Proceedings of the 2020 International Symposium on VLSI Design, Automation and Test (VLSI-DAT), Hsinchu, Taiwan, 10–13 August 2020.
66. El Zarif, N.; Hemmat, M.A.; Dupuis, T.; David, J.P.; Savaria, Y. Polara-Keras2c: Supporting Vectorized AI Models on RISC-V Edge Devices. *IEEE Access* **2024**, *12*, 171836–171852. [CrossRef]
67. Manor, E.; Greenberg, S. Custom Hardware Inference Accelerator for TensorFlow Lite for Microcontrollers. *IEEE Access* **2022**, *10*, 73484–73493. [CrossRef]
68. Lee, S.; Lee, K.B.; Joo, S.; Ahn, H.K.; Lee, J.; Kim, D.; Ham, B.; Jung, S.O. SIF-NPU: A 28nm 3.48 TOPS/W 0.25 TOPS/mm2 CNN Accelerator with Spatially Independent Fusion for Real-Time UHD Super-Resolution. In Proceedings of the ESSCIRC 2022—IEEE 48th European Solid State Circuits Conference (ESSCIRC), Milan, Italy, 19–22 September 2022; pp. 97–100.
69. Qi, M.; Ren, Y.; Sun, T.; Xu, R.; Lv, Z.; Zhou, Y.; Han, S.-T. Self-powered artificial vibrissal system with anemotaxis behavior. *Sci. Adv.* **2025**, *11*, eadt3068. [CrossRef]
70. Texas Instruments Inc. TMS320F28P55x Real-Time Microcontrollers: SPRSP85C[Z]. Revised June 2025. Available online: https://www.ti.com/lit/ds/sprsp85c/sprsp85c.pdf (accessed on 1 April 2024).
71. Kena, X.; Wei, H.; Mengke, S.; Wenjun, L.; Tianpei, L. A real-time task scheduling algorithm for multicore embedded systems. In Proceedings of the 2015 Chinese Automation Congress (CAC), Wuhan, China, 27–29 November 2015; pp. 1165–1170.
72. Zhu, W.; Wu, W.; Yang, X.; Zeng, G. TSSA: Task structure-aware scheduling of energy-constrained parallel applications on heterogeneous distributed embedded platforms. *J. Syst. Archit.* **2022**, *132*, 102741. [CrossRef]
73. Zhang, M.; Liu, L.Q.; Li, C.Z.; Wang, H.F.; Li, M. A Particle Swarm Optimization Method for AI Stream Scheduling in Edge Environments. *Symmetry* **2022**, *14*, 2565. [CrossRef]
74. Rizvandi, N.B.; Taheri, J.; Zomaya, A.Y. Some observations on optimal frequency selection in DVFS-based energy consumption minimization. *J. Parallel Distrib. Comput.* **2011**, *71*, 1154–1164. [CrossRef]
75. Fettes, Q.; Clark, M.; Bunescu, R.; Karanth, A.; Louri, A. Dynamic Voltage and Frequency Scaling in NoCs with Supervised and Reinforcement Learning Techniques. *IEEE Trans. Comput.* **2019**, *68*, 375–389. [CrossRef]
76. Huang, H.; Lin, M.; Yang, L.T.; Zhang, Q. Autonomous Power Management with Double-Q Reinforcement Learning Method. *IEEE Trans. Ind. Inform.* **2020**, *16*, 1938–1946. [CrossRef]
77. Chen, F.; Yu, H.; Jiang, W.; Ha, Y. Quality Optimization of Adaptive Applications via Deep Reinforcement Learning in Energy Harvesting Edge Devices. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2022**, *41*, 4873–4886. [CrossRef]
78. Chen, T.Q.; Moreau, T.; Jiang, Z.H.; Zheng, L.M.; Yan, E.; Cowan, M.; Shen, H.; Wang, L.; Hu, Y.; Ceze, L.; et al. TVM: An Automated End-to-End Optimizing Compiler for Deep Learning. In Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI), Carlsbad, CA, USA, 8–10 October 2018; pp. 579–594.
79. Sun, W.B.; Katsifodimos, A.; Hai, R. Accelerating machine learning queries with linear algebra query processing. *Distrib. Parallel Databases* **2025**, *43*, 8. [CrossRef]
80. Meddeb, M.; Cagnazzo, M.; Pesquet-Popescu, B. Region-of-interest-based rate control scheme for high-efficiency video coding. *APSIPA Trans. Signal Inf. Process.* **2014**, *3*, e16. [CrossRef]
81. Devindi, I.; Liyanage, S.; Jayarathna, T.; Alawatugoda, J.; Ragel, R. A novel ECG compression algorithm using Pulse-Width Modulation integrated quantization for low-power real-time monitoring. *Sci. Rep.* **2024**, *14*, 17162. [CrossRef] [PubMed]
82. Abdel-Kader, R.F.; El-Sayad, N.E.; Rizk, R.Y. Efficient Noise Reduction System in Industrial IoT Data Streams. In Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2021, Cairo, Egypt, 11–13 December 2022; pp. 219–232.
83. Ali, I.; Wassif, K.; Bayomi, H. Dimensionality reduction for images of IoT using machine learning. *Sci. Rep.* **2024**, *14*, 7205. [CrossRef]
84. Ungureanu, V.-I.; Negirla, P.; Korodi, A. Image-Compression Techniques: Classical and "Region-of-Interest-Based" Approaches Presented in Recent Papers. *Sensors* **2024**, *24*, 791. [CrossRef]
85. Pioli, L.; de Macedo, D.D.J.; Costa, D.G.; Dantas, M.A.R. Intelligent Edge-powered Data Reduction: A Systematic Literature Review. *ACM Comput. Surv.* **2024**, *56*, 234. [CrossRef]
86. Pazmiño Ortiz, L.A.; Maldonado Soliz, I.F.; Guevara Balarezo, V.K. Advancing TinyML in IoT: A Holistic System-Level Perspective for Resource-Constrained AI. *Future Internet* **2025**, *17*, 257. [CrossRef]
87. Crafton, B.; Wan, Z.; Spetalnick, S.; Yoon, J.-H.; Wu, W.; Tokunaga, C.; De, V.; Raychowdhury, A. Improving compute in-memory ECC reliability with successive correction. In Proceedings of the 59th ACM/IEEE Design Automation Conference, San Francisco, CA, USA, 10–14 July 2022; pp. 745–750.
88. Shalan, M.; Hasan, M.R.; Bai, Y.; Li, J. Enhancing Smart Home Security: Blockchain-Enabled Federated Learning with Knowledge Distillation for Intrusion Detection. *Smart Cities* **2025**, *8*, 35. [CrossRef]
89. Distributors, M.-H. Case Studies: Successful HVAC System Implementations in 2025. Available online: https://www.marhy.com/2025-hvac-success-smart-efficient-and-sustainable (accessed on 1 May 2025).

90. GigaDevice. How the High-Performance GD32G5 Series MCUs Are Driving Transformation in Digital Energy, Motor Control, and Optical Communications. Available online: https://www.gigadevice.com.cn/about/news-and-event/blog/high-performance-gd32g5-series-mcus (accessed on 1 June 2025).
91. Huang, Z.; Herbozo Contreras, L.F.; Leung, W.H.; Yu, L.; Truong, N.D.; Nikpour, A.; Kavehei, O. Efficient Edge-AI Models for Robust ECG Abnormality Detection on Resource-Constrained Hardware. *J. Cardiovasc. Transl. Res.* **2024**, *17*, 879–892. [CrossRef] [PubMed]
92. Zhou, H.; Huang, F.; Lou, W.; Gu, Q.; Ye, Z.; Hu, H.; Zhang, X. Yield prediction through UAV-based multispectral imaging and deep learning in rice breeding trials. *Agric. Syst.* **2025**, *223*, 104214. [CrossRef]
93. Pedroso, A.F.; Khera, R. Leveraging AI-enhanced digital health with consumer devices for scalable cardiovascular screening, prediction, and monitoring. *npj Cardiovasc. Health* **2025**, *2*, 34. [CrossRef]
94. Antoun, I.; Abdelrazik, A.; Eldesouky, M.; Li, X.; Layton, G.R.; Zakkar, M.; Somani, R.; Ng, G.A. Artificial intelligence in atrial fibrillation: Emerging applications, research directions and ethical considerations. *Front. Cardiovasc. Med.* **2025**, *12*, 1596574. [CrossRef] [PubMed]
95. Zhao, Y.F.; Chaw, J.K.; Ang, M.C.; Tew, Y.; Shi, X.Y.; Liu, L.; Cheng, X. A safe-enhanced fully closed-loop artificial pancreas controller based on deep reinforcement learning. *PLoS ONE* **2025**, *20*, e0317662. [CrossRef] [PubMed]
96. Venkateswara, S.M.; Padmanabhan, J. Deep learning based agricultural pest monitoring and classification. *Sci. Rep.* **2025**, *15*, 8684. [CrossRef] [PubMed]
97. Li, L.M.; Zhang, L.; Zhang, Y.F. The Intelligent Design of Solar LED Street Lamps Based on MCU. In Proceedings of the 2nd International Conference on Materials and Products Manufacturing Technology (ICMPMT 2012), Guangzhou, China, 22–23 September 2012; pp. 2005–2008.
98. Ng, N.; Souza, A.; Diggavi, S.; Suri, N.; Abdelzaher, T.; Towsley, D.; Shenoy, P. Collaborative Inference in Resource-Constrained Edge Networks: Challenges and Opportunities. In Proceedings of the 2024 Military Communications Conference, Washington, DC, USA, 28 October–1 November 2024.
99. Wang, L.F.; Li, W.Z.; Zhou, Z.D.; An, J.J.; Ye, W.; Li, Z.; Gao, H.H.; Hu, H.Y.; Liu, J.; Chen, X.M.; et al. A near-threshold memristive computing-in-memory engine for edge intelligence. *Nat. Commun.* **2025**, *16*, 5897. [CrossRef]
100. Ding, C.; Ren, Y.; Liu, Z.; Wong, N. Transforming memristor noises into computational innovations. *Commun. Mater.* **2025**, *6*, 149. [CrossRef]
101. Immonen, R.; Hamalainen, T. Tiny Machine Learning for Resource-Constrained Microcontrollers. *J. Sens.* **2022**, *2022*, 7437023. [CrossRef]
102. Lin, J.; Chen, W.-M.; Lin, Y.; Cohn, J.; Gan, C.; Han, S. MCUNet: Tiny Deep Learning on IoT Devices. In Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS), Electr Network, Vancouver, BC, Canada, 6–12 December 2020.
103. Wang, P. Neural Network Optimization Framework for NPU-MCU Heterogeneous Platforms. *Appl. Comput. Eng.* **2025**, *145*, 43–50. [CrossRef]
104. Liu, Y.; Qu, H.; Chen, S.; Feng, X. Energy efficient task scheduling for heterogeneous multicore processors in edge computing. *Sci. Rep.* **2025**, *15*, 11819. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.