Edge AI: Embedded Intelligence, a Review on Hardware and Applications

HANI Al-MIMI[1], AHMAD AL-DAHOUD[2], ALI AL-DAHOUD[1], MOHAMED FEZARI[3]
[1]Faculty of Science and IT,
Al-Zaytoonah University of Jordan,
Amman,
JORDAN

[2]Faculty of Architecture and Design,
Al-Zaytoonah University of Jordan,
Amman,
JORDAN

[3]Faculty of Technology,
Badji Mokhtar Annaba University,
ALGERIA

Abstract: - Edge AI brings artificial-intelligence inference from the cloud to resource-constrained devices at the network edge, enabling real-time, low-latency, and privacy-preserving decision-making. This review examines edge AI from a hardware-centric and quantitative perspective. We introduce a unified analytical framework that models inference latency, bandwidth, energy, computational complexity, throughput, and model compression, and use it to compare cloud versus edge execution. We benchmark the principal classes of AI hardware accelerators (VPU, GPU, NPU, and TPU) using public performance and efficiency figures, and present an explicit latency- and energy-aware placement algorithm. We further review representative real-world applications, the main deployment challenges, and future directions including federated learning, neuromorphic computing, and 6G-assisted inference.

Key-Words: - Edge AI, Artificial intelligence, Cloud computing, Internet of Things, IoT, Machine learning, Deep learning, Performance modelling, Hardware accelerators.

Received: January 19, 2026. Revised: March 25, 2026. Accepted: May 27, 2026. Published: September 3, 2026.

1 Introduction

Artificial intelligence (AI) has many important applications, such as natural language processing (NLP) [1], [2], the Internet of Things [3], [4], mobile robotics [5], and edge AI [6]. Edge AI enables real-time decision-making, low latency, and improved privacy, [6]. However, companies adopting edge AI must develop scalable, cost-effective solutions that are simple to deploy across diverse applications, while also addressing the hardware challenge of providing sufficient processing power in resource-constrained environments. Edge computing requires local processing of data at the network's edge, closer to where it is generated. This could be on devices like smartphones, sensors, cameras, or even within industrial machinery. Faster response times and less dependency on network connectivity are made possible by edge devices' ability to analyze and act upon raw data locally rather than sending it to a centralized cloud for processing.

The market for edge AI is expanding quickly in the US and Europe because of strict data protection laws and regulations. For local data processing and to ensure compliance with the General Data Protection Regulation (GDPR), European businesses are investing in edge AI. Aiming to guarantee reliable AI innovation, the EU AI Act also has an impact on the creation and application of edge AI solutions in a number of industries.

This review examines the principal advantages and limitations of deploying AI at the edge, the techniques available for overcoming the computational and memory constraints of embedded hardware, and the approaches used to strengthen the

robustness, safety, security, and dependability of edge AI systems. It further compares the leading classes of edge AI hardware accelerators and situates edge intelligence in relation to adjacent paradigms such as distributed and swarm intelligence. Beyond this qualitative discussion, the review supplies the quantitative apparatus required to address these issues rigorously.

The application of AI platforms and solutions near the end user's environment, on the edge of a network, is known as edge AI. Devices like edge computers, Internet of Things (IoT) hardware, and small, local data centers are commonly used for edge AI implementation, [7]. Edge AI use cases allow AI computations and inferences to be processed on edge servers, gateways, and IoT devices placed closer to the user rather than at a huge, centralized data center.

In this sense, data produced by smart cameras, sensors, and other IoT devices can still be analyzed within the device even in the event that Internet connectivity is not available. Central Processing Units (CPUs), Neural Processing Units (NPUs), and microcontrollers are just a few examples of the hardware that can be used to develop AI at the edge. In an edge AI ecosystem, ML models operate directly on the edge devices while data is processed locally. These edge AI/ML devices monitor, gather, and process device data using embedded models. The data can then be used by businesses to predict future events, fix mistakes, and optimize business operations.

The processing flow of a conventional cloud-based AI application, in which raw data are transmitted from the end devices to a centralized data center for inference and the results are returned over the network, is shown in Figure 1.

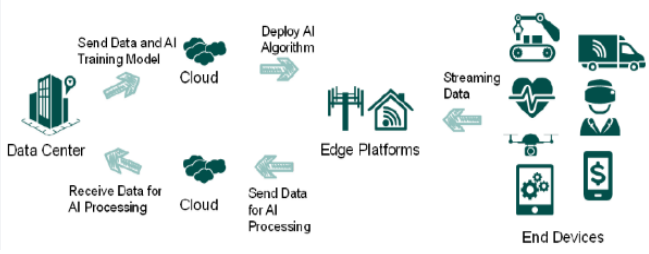

Fig. 1: Processing flow of a cloud-based AI use case. Raw data generated by the end devices are transmitted over the network to a centralized data center, where AI inference is performed; the results are then returned to the devices, incurring wide-area backhaul latency and bandwidth cost.
Source: created by the authors.

By contrast, the edge AI processing flow, in which inference is executed locally on edge platforms and gateways adjacent to the data source so that only compact results are exchanged with the cloud when required, is shown in Figure 2.

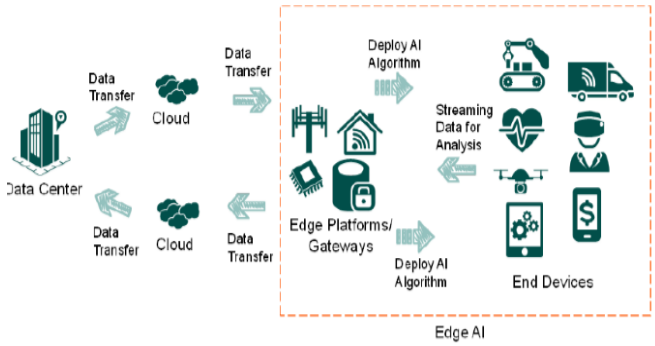

Fig. 2: Processing flow of an edge AI deployment. AI models are deployed on edge platforms and gateways located close to the end devices, so that inference is executed locally on the incoming sensor, image, or audio data and only compact results are exchanged with the cloud when necessary. Compared with the cloud flow of Figure 1, this eliminates the backhaul hop, reducing latency and bandwidth consumption while keeping raw data on the device.
Source: created by the authors.

Deploying edge AI offers businesses numerous observable advantages, such as the following, [7]:

- Lower bandwidth consumption and reduced network latency: Neural networks experience latency when they transfer massive amounts of data to and from a remote data center. Through local processing at the device/platform level, edge AI computing can lower network latency. Consequently, this reduces network bandwidth costs.
- Real-time analytics: Businesses can do real-time information analysis at the source device or platform by utilizing edge AI's high-performance features.
- Less electricity consumption: Devices need less electricity to work since AI reduces the amount of data that must be transferred at the edge.
- Better business outcomes: Businesses benefit from faster decision-making with edge AI solutions. Enterprise users of edge AI will experience a decrease in operating costs and an increase in efficiency when operational data can be analyzed rapidly.

We will examine four businesses that use edge AI to better understand its practical applications. The rest of the paper is organized as follows. Section 2 describes the power of AI at the edge, showing its advantages and capabilities. Section 3 lists the main differences between cloud AI and edge AI. Section 4 illustrates how edge AI works and presents some edge AI technologies and applications. Section 5 introduces a unified system model and performance analysis for edge AI, formalising latency, bandwidth, energy, complexity, throughput, compression, and the cloud-versus-edge placement decision. Section 6 shows the main challenges that face edge AI. Section 7 demonstrates the main hardware accelerators used for edge AI together with a quantitative benchmark comparison. Section 8 presents some real-world applications and use cases. Section 9 gives a discussion on edge AI and its future directions. Section 10 concludes the paper.

1.1 Contributions and Relation to Existing Surveys

Several recent surveys map the edge-AI landscape from a taxonomy or systems perspective, [6], [7], [8], [9]. This review is deliberately *hardware-centric* and *quantitative*. Its distinct contributions are: (i) a unified, closed-form performance model that spans inference latency, bandwidth, energy, computational complexity, throughput, and model compression within a single notation (Section 5); (ii) a quantitative cross-accelerator benchmark of VPU, GPU, NPU, and TPU classes using public TOPS, power, and efficiency figures (Section 7); and (iii) an explicit latency/energy-aware placement algorithm that links these models to the cloud-versus-edge decision (Algorithm 1). Table 3 (Appendix) positions this work against representative recent surveys along these dimensions. The authors' earlier publications cited here, namely [4], [5], and [10], are used only as background and application context; the analytical framework of Section 5, the quantitative benchmark of Section 7, and the placement algorithm of Algorithm 1 are original to this review and do not reproduce material from those works. The text has been prepared specifically for this article, and the authors have verified its originality using institutional similarity-checking software prior to submission.

2 The Power of AI at the Edge

When AI is integrated with edge computing, it unlocks a range of transformative capabilities, [7]:

- *Decreased Latency*: Edge AI reduces the amount of time needed to analyze data and react to events by processing data locally. This is essential for applications like smart grid management, industrial automation, and driverless cars that require real-time reactions.
- *Improved Security and Privacy*: By processing sensitive data at the edge, less data must be sent across networks, lowering the possibility of data breaches and guaranteeing adherence to privacy laws.
- Improved Bandwidth Efficiency: By minimizing the volume of data that must be sent to the cloud, edge AI dramatically lowers bandwidth usage and related expenses. This also enables companies or individuals to add more sensors and other relevant items without having to make significant infrastructure investments.
- *Increased Reliability and Resilience*: Edge devices can operate independently even when connectivity to the cloud is disrupted, ensuring continuous operation in critical applications.
- *Lowered Costs*: Edge AI can help organizations significantly lower their operational costs by minimizing the need for wide data transfer and cloud processing.

Each of these qualitative advantages is given a precise, measurable form in Section 5: decreased latency corresponds to Equations (1)–(5), bandwidth efficiency to Equations (6)–(7), lowered energy and cost to Equations (8)–(11), and the accuracy/size/latency trade-offs behind on-device deployment to Equations (16)–(21). Figure 3 displays the main advantages of edge computing for real-world applications.

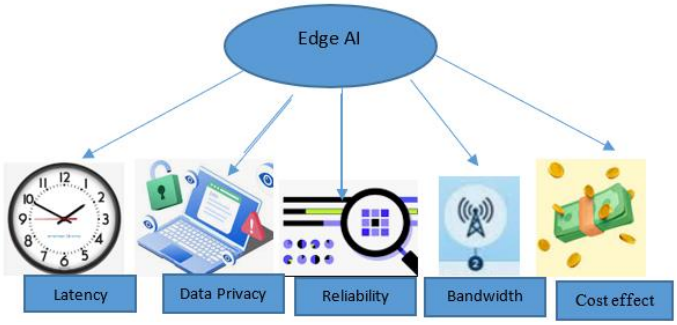

Fig. 3: Principal advantages of edge computing for real-world applications. Specifically: reduced latency, lower bandwidth consumption, improved privacy and security, increased reliability under intermittent connectivity, and greater energy efficiency. Source: created by the authors.

3 The Main Differences Between Edge AI and Cloud AI

In cloud AI, a feature performed on remote servers is also performed on local devices. The latency is higher in the cloud due to network delay and lower at the edge as computation is local. Cloud operations require constant internet access, whereas edge devices can operate offline. Cloud data transport is comparatively hazardous, while data that stays on the device is safer. High energy consumption resulting from extensive data transfer is reduced at the edge, since less data is sent, [11]. These qualitative contrasts are formalised in Section 5, where Equation (3) adds the wide-area backhaul term that distinguishes cloud from edge latency, and Equation (5) expresses the resulting speedup.

By facilitating intelligent, real-time decision-making in settings where dependability and speed are crucial, edge AI is revolutionizing entire industries. The main advantages of edge AI are summarised once here and referred to thereafter, [7]: low latency and real-time processing, since data processing takes place locally; reduced bandwidth usage, because only pertinent insights are transmitted; improved privacy and security, by eliminating the need to transfer sensitive data to external servers; enhanced reliability, since edge devices keep operating under spotty connectivity; and energy efficiency, by locally processing only the data that is required.

4 How Edge AI Works

Edge AI integrates machine learning models into resource-constrained devices. These crucial steps are followed in the process:

1. Model Training. Large datasets are first used to train AI models in a robust computing environment, either the cloud or a dedicated data center. High processing power is needed for this training phase.
2. Model Optimization. Trained models must be optimized and compressed because of the resource constraints of edge devices. Methods like knowledge distillation, quantization, and pruning are employed to decrease the size of the model without appreciably sacrificing accuracy. Section 5.6 formalises these methods through the compression ratio, the quantized memory footprint, and the distillation objective in Equations (16)–(19).
3. Model Deployment on Edge Devices. After optimization, the model is implemented on edge hardware, including microcontrollers, mobile devices, and IoT sensors. Well-known frameworks such as PyTorch Mobile, ONNX Runtime, and TensorFlow Lite make this deployment possible.
4. Inference and Decision Making. In order to produce predictions and judgments in real time, the model continuously evaluates incoming data (such as sound, pictures, or sensor readings) while it is operating on an edge device.

4.1 The Main Technologies Enabling Edge AI

Edge AI is made possible by a number of technologies, such as, [8]:

Hardware Accelerators: FPGAs and ASICs, Apple Neural Engine, Google Coral, Intel Movidius, NVIDIA Jetson, and AI chips and processors that are specially made for high-performance edge AI computing.

Software Frameworks: TensorFlow Lite, a condensed version of TensorFlow tailored for mobile and embedded devices, is one of the edge AI frameworks. PyTorch Mobile is a framework designed for mobile devices that run AI models. Interoperability between various AI models and hardware is made possible by the ONNX Runtime.

Integration of 5G and IoT: 5G networks offer fast, low-latency connections, which make it possible for edge AI devices to easily interface with cloud-based systems as necessary. Additionally, IoT devices serve

as data sources, providing edge AI systems with real-time data for intelligent processing.

4.2 Edge AI Applications

Numerous industries are already being significantly impacted by 5G and edge AI, such as [8], [9]:

- Autonomous Vehicles: Edge AI gives self-driving cars the ability to evaluate sensor data locally, make rapid decisions, and prevent collisions.
- Healthcare and Medical Devices: By using edge AI to analyze vital indicators in real-time, wearable health monitors can notify physicians of possible health issues before crises arise.
- Smart Manufacturing: Edge AI is used by factories for process optimization, real-time quality control, and predictive maintenance.
- Retail and Customer Experience: AI-powered cameras in retailers may identify inventory shortages, evaluate consumer behavior, and provide individualized shopping experiences.
- Security and Surveillance: Without requiring cloud-based processing, edge AI-powered cameras are able to recognize faces, identify suspicious activity, and sound an alarm.

Figure 4 presents some of the edge AI applications.

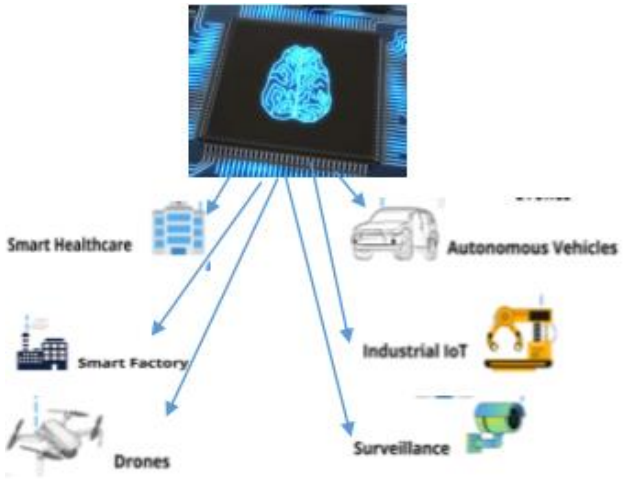

Fig. 4: Representative application domains of edge AI. These include autonomous vehicles, healthcare and wearable monitoring devices, smart manufacturing, retail analytics, and security and surveillance, each of which relies on low-latency local inference.
Source: created by the authors.

5 System Model and Performance Analysis

This section introduces a compact analytical framework for reasoning quantitatively about edge AI. The same notation is used throughout: D_{in} and D_{out} denote the input and output data size (bits); C denotes the workload (CPU cycles, or multiply–accumulate operations where stated); f_{dev}, f_{srv}, and f_{cloud} denote the device, edge-server, and cloud clock rates (cycles · s^{-1}); $R_{\uparrow}$ and $R_{\downarrow}$ are the uplink and downlink rates (bit · s^{-1}); P_{tx} and P_{idle} are the transmit and idle power (W); κ is the effective switched capacitance; b is the quantization bit-width; and N is the number of model parameters.

5.1 Inference Latency: Local, Edge, and Cloud

For local (on-device) execution, the inference latency is the workload divided by the device clock:

$$T_{\text{loc}} = \frac{C}{f_{\text{dev}}} \tag{1}$$

When the task is offloaded to a nearby edge server, the latency is the sum of the uplink transmission delay, the remote computation delay, and the downlink delay for returning the result:

$$T_{\text{edge}} = \frac{D_{\text{in}}}{R_{\uparrow}} + \frac{C}{f_{\text{srv}}} + \frac{D_{\text{out}}}{R_{\downarrow}} \tag{2}$$

Cloud execution adds a wide-area backhaul term T_{wan} between the access point and the data center:

$$T_{\text{cloud}} = \frac{D_{\text{in}}}{R_{\uparrow}} + T_{\text{wan}} + \frac{C}{f_{\text{cloud}}} + \frac{D_{\text{out}}}{R_{\downarrow}} \tag{3}$$

The transmission rate follows the Shannon–Hartley capacity, which links latency to channel quality (bandwidth B, channel gain g, noise spectral density N_0):

$$R_{\uparrow} = B \log_2\left(1 + \frac{P_{\text{tx}}\, g}{N_0 B}\right) \tag{4}$$

The inference acceleration of edge over cloud execution, the single most informative figure of merit, is the latency speedup:

$$S = \frac{T_{\text{cloud}}}{T_{\text{edge}}}, \qquad \Delta T \approx T_{\text{wan}} + C\left(\frac{1}{f_{\text{cloud}}} - \frac{1}{f_{\text{srv}}}\right) \tag{5}$$

Equation (5) makes the paper's central claim precise: edge execution wins whenever the eliminated backhaul latency T_{wan} (plus queueing) exceeds the computational disadvantage of the less powerful edge server. This is the analytical basis for the real-time benefits discussed in Sections 2 and 3, [12].

5.2 Bandwidth Consumption

The bandwidth advantage of edge AI follows from transmitting inferences or compact features rather than raw data. The data-reduction ratio is

$$\rho = 1 - \frac{D_{\text{out}}}{D_{\text{in}}} \tag{6}$$

For a deployment of K devices, each generating an input of size D_{in} at a rate of λ events · s^{-1}, the aggregate uplink load under the cloud and edge paradigms is

$$\Phi_{\text{cloud}} = K\lambda D_{\text{in}}, \qquad \Phi_{\text{edge}} = K\lambda D_{\text{out}} = (1-\rho)\,\Phi_{\text{cloud}} \tag{7}$$

For example, a 1080p camera stream of roughly 3 Mbit · s^{-1} reduces to a few hundred bytes of detection metadata once inference runs locally, a reduction ratio approaching unity, which quantifies the bandwidth savings asserted in Section 2, [11], [12].

5.3 Energy and Power Efficiency

Using the standard CMOS dynamic-power relation $P_{\text{dyn}} = \kappa f^3$, the device energy for a local inference is:

$$E_{\text{loc}} = P_{\text{dyn}}\, T_{\text{loc}} = \kappa\, C\, f_{\text{dev}}^2 \tag{8}$$

When offloading, the device expends transmit energy during upload and idle energy while awaiting the remote result:

$$E_{\text{edge}} = P_{\text{tx}} \frac{D_{\text{in}}}{R_{\uparrow}} + P_{\text{idle}} \frac{C}{f_{\text{srv}}} \tag{9}$$

At the hardware level, power efficiency is expressed as operations per watt, and the energy per inference follows directly:

$$\eta = \frac{\text{Throughput}}{\text{Power}} \left[\text{TOPS} \cdot \text{W}^{-1}\right] \tag{10}$$

$$E_{\text{inf}} = \frac{P}{\text{inferences} \cdot \text{s}^{-1}} \tag{11}$$

Because Equation (8) scales energy quadratically with the clock frequency, dynamic voltage and frequency scaling is fundamental to edge devices: halving the clock frequency quarters the energy at the cost of doubling latency, the precise trade-off behind the "less electricity consumption" benefit of Section 2, [13].

5.4 Computational Complexity

For a convolutional layer with input height H_{in}, padding p, kernel size k, and stride s, the output spatial dimension is:

$$H_{\text{out}} = \left\lfloor \frac{H_{\text{in}} + 2p - k}{s} \right\rfloor + 1 \tag{12}$$

The multiply–accumulate (MAC) count of that layer, for C_{in} input and C_{out} output channels, and the equivalent floating-point operation count, are:

$$\text{MAC}_{\text{conv}} = H_{\text{out}} W_{\text{out}}\, C_{\text{out}}\, (k^2 C_{\text{in}}), \qquad \text{FLOPs} \approx 2\,\text{MAC} \tag{13}$$

The total complexity of a model is the sum over its layers (a fully connected layer contributes $\text{MAC}_{\text{fc}} = N_{\text{in}} N_{\text{out}}$):

$$\text{FLOPs}_{\text{model}} = \sum_{\ell} 2\ \text{MAC}_{\ell} \tag{14}$$

These expressions place the "model optimization" requirement of Section 4 on a measurable footing. For reference, MobileNetV2 requires about 0.3 GFLOPs per inference, ResNet-50 about 4 GFLOPs, and DenseNet-161 about 15.6 GFLOPs, which is why compression is mandatory before deployment on the accelerators of Section 7, [14], [15].

5.5 Throughput: The Roofline Model

The attainable throughput of an accelerator is bounded by either its peak compute or its memory bandwidth, as a function of the arithmetic (operational) intensity $I = \text{FLOPs/Bytes}$:

$$P_{\text{attain}} = \min\left(P_{\text{peak}},\ I \cdot BW_{\text{mem}}\right) \tag{15}$$

The ridge point $I^* = P_{\text{peak}}/BW_{\text{mem}}$ separates the memory-bound regime ($I < I^*$) from the compute-bound regime, [16]. Plotting the VPU, GPU, NPU, and TPU of Section 7 together with representative models on these axes explains, at a glance, why different accelerators suit different workloads: a small, low-intensity model such as MobileNet is typically memory-bound, whereas a high-intensity model such as ResNet-50 can reach the compute roof, [17], [18].

5.6 Model Compression

The benefit of compression is captured by the compression ratio and the quantized memory footprint:

$$\mathrm{CR} = \frac{\mathrm{size}_{\mathrm{original}}}{\mathrm{size}_{\mathrm{compressed}}}, \tag{16}$$

$$M_{\mathrm{quant}} = \frac{N \cdot b}{8} \text{ bytes} \tag{17}$$

For uniform quantization, the worst-case weight error decays geometrically in the bit-width b (parameter range R, dimension D):

$$\| \theta - \hat{\theta} \|_2 \leq \frac{R\sqrt{D}}{2^b} \tag{18}$$

Knowledge distillation trains a compact student to imitate a larger teacher, with temperature T, mixing weight α, and student/teacher logits z_s, z_t, the objective combines a hard-label and a soft-label term:

$$\mathcal{L} = \alpha\, \mathcal{L}_{\mathrm{CE}}\big(y, \sigma(z_s)\big) + (1 - \alpha)\, T^2\, \mathrm{KL}\big(\sigma(z_t/T) \,\|\, \sigma(z_s/T)\big) \tag{19}$$

These models quantify documented results: Deep Compression shrank AlexNet by roughly 35× (240 MB to 6.9 MB) with negligible accuracy loss, [19], and a distilled model such as DistilBERT is about 40% smaller and 60% faster while retaining about 97% of the teacher's accuracy, [14].

5.7 Memory–Performance Trade-off and Deployment Constraint

The defining constraint of on-device inference is that the working set must fit within the device memory:

$$M_{\mathrm{weights}} + M_{\mathrm{activations}} + M_{\mathrm{runtime}} \leq M_{\mathrm{device}} \tag{20}$$

Deployment is therefore a constrained optimization, selecting a compression and placement configuration Θ that minimizes a weighted latency–energy cost subject to an accuracy floor and the memory bound (β_T, β_E are weights):

$$\min_{\theta} \beta_T\, T(\theta) + \beta_E\, E(\theta) \quad \text{s.t.} \quad \mathrm{Acc}(\theta) \geq \mathrm{Acc}_{\min},\ M(\theta) \leq M_{\mathrm{device}} \tag{21}$$

Equation (21) unifies the accuracy-versus-size-versus-latency trade-offs that recur throughout this review into a single objective and motivates the hardware restrictions and model-optimization trade-offs listed in Section 6.

5.8 The Cloud-versus-Edge Placement Decision

Combining the latency and energy models, a task should be offloaded only when doing so is both faster and no more energy-costly than local execution:

$$\text{Decision} = \begin{cases} \text{offload}, & T_{\mathrm{edge}} < T_{\mathrm{loc}} \ \wedge\ E_{\mathrm{edge}} \leq E_{\mathrm{loc}} \\ \text{local}, & \text{otherwise} \end{cases} \tag{22}$$

Algorithm 1 operationalises this rule using the weighted cost of Equation (21) and the memory constraint of Equation (20). It can be extended to partitioned inference, where a deep network is split at a layer cut-point, and its head executes on-device while its tail executes on the server, which is the current frontier of collaborative edge intelligence, [12].

Algorithm 1 — Latency/energy-aware inference placement

Input: task (D_in, C, D_out); device (f_dev, P_tx, P_idle);
link (R_up, R_down); server f_srv; weights (beta_T, beta_E)

```
T_loc  <- C / f_dev                          # Eq. (1)
E_loc  <- kappa * C * f_dev^2                # Eq. (8)
T_edge <- D_in/R_up + C/f_srv + D_out/R_down # Eq. (2)
E_edge <- P_tx*(D_in/R_up) + P_idle*(C/f_srv) # Eq. (9)
J_loc  <- beta_T*T_loc  + beta_E*E_loc       # Eq. (21)
J_edge <- beta_T*T_edge + beta_E*E_edge
if J_edge < J_loc and memory_fits(model):    # Eq. (20)
    return OFFLOAD
else:
    return LOCAL
```

Output: placement decision in {LOCAL, OFFLOAD}

6 Challenges of Edge AI

There are many challenges that face edge AI, such as [9]:

- Hardware Restrictions: The processing speed, memory, and battery life of edge devices are constrained, which is precisely the feasibility region defined by Equations (20)–(21).

- Model Optimization Trade-offs: Accuracy and performance may suffer if the model's size is decreased; the quantization-error bound of Equation (18) makes this trade-off explicit.
- Data Privacy Regulations: It might be difficult to ensure adherence to HIPAA, GDPR, and other data privacy rules.
- Integration Complexity: It takes strong software engineering to integrate edge AI with current cloud infrastructures and IoT devices.

7 The Necessity of Hardware Accelerators for Edge AI

Specialized AI hardware accelerators are needed to perform complicated machine learning operations on edge devices. These accelerators increase speed and performance while providing increased scalability, maximum security, dependability, and effective data management. The following accelerators are the principal options, [10], [14], [15], [17]:

- Vision Processing Unit (VPU): A VPU is a microprocessor designed to speed up AI and machine learning algorithms for vision. Like a video processing unit used with neural networks, it enables tasks such as image processing and balances edge AI workloads with great efficiency, operating with high-performance precision and minimal power consumption.
- Graphics Processing Unit (GPU): A GPU is an electronic circuit that can generate graphics for a display. GPUs handle several data sets at once, which makes them well suited to machine learning, video editing, and gaming, and they are widely used in workstations, gaming consoles, tablets, and smartphones owing to their capacity to perform intricate machine learning jobs.
- Neural Processing Unit (NPU): NPUs gained popularity first on mobile platforms because power efficiency is central to their design. NPUs manage AI inference workloads with low energy use, ensuring long battery life in devices such as smartphones and IoT devices. GPUs, despite their greater throughput, generally consume much more energy, which makes them more appropriate for workstations and data centers where power limitations are less important.
- Tensor Processing Unit (TPU): Google's TPU is an ASIC for executing neural-network machine learning algorithms. It runs efficiently and consumes less energy, and the Google Cloud Platform with TPUs is an excellent option for machine learning applications that do not need extensive cloud infrastructure.

Table 1 (Appendix) provides a qualitative comparative overview of VPUs, GPUs, NPUs, and TPUs across various aspects relevant to their design and application. Table 2 (Appendix) then supplements this with quantitative benchmark figures. The best choice of processor depends heavily on the specific workload, performance requirements, power constraints, cost considerations, and the software ecosystem being used.

Worked example. Consider MobileNetV2 at roughly 0.3 GFLOPs per inference (Equation 14). Dividing each accelerator's peak throughput from Table 2 (Appendix) by this workload gives a theoretical upper bound on frame rate; comparing that bound against the measured rate (for the Edge TPU, about 400 fps) exposes the efficiency gap caused by memory-bandwidth limits and host overhead, exactly the behaviour predicted by the roofline model of Equation (15). Reporting such predicted-versus-measured comparisons, rather than peak TOPS alone, is the recommended way to benchmark edge accelerators.

8 Real-World Applications of Edge AI

Some of the real-world applications of edge AI are listed below, [8]:

- Autonomous Vehicles: Edge AI helps self-driving cars make navigational judgments, avoid obstacles, and recognize objects in real time.
- Industrial Automation: By facilitating quality assurance, predictive maintenance, and real-time production-line modifications, edge AI streamlines manufacturing.
- Smart Cities: Edge AI powers smart streetlights, traffic management, and environmental monitoring, making cities more efficient and sustainable.
- Healthcare: Real-time analysis of patient data via wearable technology is made possible by edge AI, speeding up diagnosis and enhancing patient care.

- Retail: Edge AI can customize client experiences, optimize inventory management, and improve store operations through real-time insights.

The following are some examples of applications and use cases for edge AI, [12]:

A) Supermarkets. Sensors and camera-vision images are used by supermarkets to gather and examine data on empty shelves. To improve the consumer experience, the edge AI application alerts store employees about stockouts on the floor aisle. Several retailers are also investing in facial recognition and eye scanning to deter shoplifting in light of the growth in retail theft. As an illustration, the partnership between computer-vision startup Deep North and NVIDIA shows how AI is giving retail a competitive edge: Deep North's software, based on NVIDIA's EGX computing platform, helps malls and physical retail establishments enhance the in-store experience using deep-learning algorithms and image analysis from shop cameras, [13]. Important outcomes of this deployment include heatmaps of stores and customer traffic, identification of high-wait checkout periods, and customer-demographic and conversion metrics.

B) Supply Chain. Edge AI computing can be used by supply-chain managers to forecast and control inventory levels. Data from RFID tags, GPS sensors, and other IoT edge devices is processed and analyzed by AI at the edge. Businesses use this information to spot slow-moving products and possible raw-material shortages, and to streamline shipping and logistics so as to cut delivery backlogs and optimize routes, [20]. As an example, the Taiwanese electronics maker Lanner installs edge AI equipment on trucks, subways, and ferries to remotely monitor dispersed mobile vehicles; a car-ferry operator used Lanner's V6S edge computing device for passenger safety, traffic control, onboard detection, and surveillance, where the edge neural network was essential to maintaining seamless ferry operations.

C) Healthcare. In the medical field, timely gathering of patient-specific data can be the difference between life and death. Edge AI platforms, medical devices, and other technologies enable data collection so that researchers and physicians can draw conclusions in real time and make faster decisions. Virtual patient services, remote patient monitoring, health tracking via medical equipment, and scanned-image analysis are healthcare use cases that benefit from edge AI, which also enables more reliable infrastructure, low-latency processing, lower bandwidth costs, and better patient-data protection, [21]. Laparoscopy is an apt illustration: a laparoscope equipped with edge AI offers surgeons ultra-low-latency streaming, supports the detection of irregularities, offers automated corrections, and detects bleeding instantly.

D) Smart Manufacturing. Smart manufacturing is one of the most mature industrial settings for edge AI, in which local inference is embedded directly into production equipment, making it more autonomous and responsive. In this context, increasing the "intelligence" of machines is a fundamental component of the smart factory, enabling predictive maintenance, autonomous mobility, and higher-level decision-making, [22]. Use cases include real-time quality control through AI and deep-learning models that identify flaws and reduce waste; predictive maintenance, where sensors, camera images, and trained models are analyzed to diagnose machinery malfunction early and reduce downtime; and self-healing or self-optimization, where continuous data from equipment keeps machines efficient. As an example, Advantech installed a smart-factory system for a major semiconductor manufacturer, integrating smart sensors and power meters across a facility and linking the IoT data to a back-end platform for central monitoring; this edge AI implementation achieved roughly 10% energy savings at the production facility.

9 Discussion

Edge AI has progressed from centralized, cloud-dependent processing toward distributed inference embedded in resource-constrained devices. Its development has been governed by three interacting factors: the capability of the underlying hardware, the diversity of deployed applications, and the cost of computation and data transfer. Rather than restating the benefits enumerated in Sections 2 and 3, this section synthesises the trajectory of the field along these three axes and connects the emerging trends to the analytical models introduced in Section 5.

9.1 Hardware Evolution

Early AI systems relied heavily on centralized cloud infrastructure because of the computational demands of machine-learning models. The rise of specialized hardware was a turning point: TPUs, NPUs, and neuromorphic chips have optimized AI workloads for edge environments, offering high performance at low power. The miniaturization of semiconductor technology (5 nm and 3 nm nodes) and system-on-chip (SoC) and module-on-chip (MoC) integration have compacted computational power into smaller form factors, while high-bandwidth memory and 3D chip stacking have eased the latency and bandwidth bottlenecks that Equation (15) attributes to the memory-bound regime. Emerging photonic and quantum-inspired processors hint at further leaps in speed and efficiency.

9.2 Application Diversification

Edge AI applications have expanded from simple sensor-based tasks to complex, mission-critical systems. Early use cases included smart-home devices and wearables; today's span healthcare (wearable electrocardiogram (ECG) monitors and AI diagnostic tools), autonomous systems (self-driving cars and drones relying on real-time object detection), industrial IoT (predictive maintenance), and smart cities (traffic and energy-grid optimization). The shift toward privacy-sensitive applications and latency-critical tasks such as robotic surgery has further cemented edge AI's necessity, and the integration of tiny machine learning (tinyML) has democratized AI for ultra-low-power devices such as agricultural sensors and disposable medical tools.

9.3 Cost Dynamics

The cost of edge AI has evolved along two dimensions. Early custom-ASIC hardware was expensive owing to high research and manufacturing outlays, but economies of scale, open-source frameworks such as TensorFlow Lite and PyTorch Mobile, and modular chip designs have lowered the barriers, as the Raspberry Pi and Arduino ecosystems illustrate. At the same time, edge AI reduces long-term expense by minimizing cloud dependency, data-transmission fees, and energy consumption, exactly the aggregate-bandwidth and energy savings formalised in Equations (7) and (8).

9.4 Future Directions

Several directions are poised to shape the next phase of edge AI. Each is connected below to the system model of Section 5.

Federated learning. Federated learning trains a shared model across many edge devices while keeping data local, improving privacy. Each device k minimizes a local empirical loss $F_k(w)$ on its private dataset $\mathcal{D}_k$ of size n_k, and the server optimizes the dataset-weighted global objective:

$$F(w) = \frac{1}{\sum_k n_k} \sum_{k=1}^{K} n_k \, F_k(w) \tag{23}$$

In the FedAvg scheme, each device performs local stochastic gradient descent and the server aggregates the updates by weighted averaging:

$$w_k^{t+1} = w_k^t - \eta \, \nabla F_k(w_k^t), \tag{24}$$

$$w^{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \, w_k^{t+1} \tag{25}$$

Because only model updates are exchanged, never raw data, the per-round communication cost scales roughly as $2|w|K$, which is why federated learning is fundamentally a bandwidth-versus-privacy trade-off connected to Equation (7); heterogeneity-aware variants such as FedProx and hierarchical edge aggregation reduce this cost further, [23].

Neuromorphic computing. Neuromorphic chips emulate biological neurons and process information through sparse, event-driven spikes. Because energy is consumed only on spike events, the energy of a spiking neural network scales with the spike count rather than with dense MAC operations:

$$E_{\text{SNN}} \approx N_{\text{spikes}} \cdot E_{\text{SOP}}, \qquad E_{\text{ANN}} \approx N_{\text{MAC}} \cdot E_{\text{MAC}} \tag{26}$$

Reported energy per synaptic operation is in the low picojoule range (for instance, about 12.7 pJ/SOP for a 28 nm digital design and roughly 20 pJ per event for Loihi- and TrueNorth-class chips), and recent edge spiking systems have reported one to two orders of magnitude lower inference energy than a comparable embedded GPU. Event-driven sparsity is thus the mechanism behind the energy advantage that Equation (8) only bounds for conventional hardware, [24].

6G networks. Future 6G networks target sub-millisecond air-interface latency and near-terabit peak rates. In terms of Equation (2), this shrinks the uplink term $D_{\text{in}}/R_{\uparrow}$ and the backhaul term of Equation (3),

making device–edge–cloud collaborative inference and over-the-air federated aggregation far cheaper. Rather than a separate topic, 6G is best understood as the communication layer that reduces the transmission-dependent terms of the latency and bandwidth models.

Self-learning edge AI. Devices will increasingly adapt and improve their models on-device, without the need for cloud retraining, moving edge AI toward ubiquitous, intelligent, and privacy-preserving computing embedded into everyday devices and systems.

10 Conclusion

Edge AI represents a revolutionary change in the use of artificial intelligence, merging advanced hardware innovation with real-world applications to enable intelligent decision-making at the source of data generation. By leveraging specialized hardware such as neuromorphic chips, low-power GPUs, NPUs, and TPUs, edge AI systems overcome traditional cloud-dependent limitations, offering reduced latency, enhanced privacy, and improved energy efficiency, and they empower applications across autonomous vehicles, smart healthcare, industrial IoT, and real-time surveillance.

Beyond surveying these developments, this review contributes a unified, closed-form performance framework that expresses inference latency, bandwidth, energy, computational complexity, throughput, and model compression within a single notation (Section 5); a quantitative cross-accelerator benchmark of the VPU, GPU, NPU, and TPU classes (Section 7); and an explicit latency/energy-aware placement algorithm that connects these models to the cloud-versus-edge decision (Algorithm 1). Together, these tools turn the qualitative advantages of edge AI into measurable, comparable quantities, which is the principal way this review differs from existing taxonomy- and systems-oriented surveys, as summarised in Table 3 (Appendix).

Challenges remain in balancing computational power against energy constraints, ensuring scalability, and addressing security vulnerabilities, and these define the feasibility region captured by Equations (20)–(21). As hardware continues to evolve, driven by neuromorphic, photonic, and quantum-inspired architectures, and as 6G reduces the communication terms of the latency model, edge AI is poised to redefine the boundaries of intelligent systems while unlocking new opportunities for innovation across industries.

References:

[1] T. Kanan, G. G. Kanaan, R. Al-Shalabi, and A. Aldaaja, "Offensive Language Detection in Social Networks for Arabic Language Using Clustering Techniques," *International Journal of Advances in Soft Computing & Its Applications*, vol. 13, no. 2, 2021.

[2] D. Aqel and B. Hawashin, "Arabic relative clauses parsing based on inductive logic programming," *Recent Patents on Computer Science*, vol. 11, no. 2, pp. 121–133, 2018.

[3] M. A. Shenify, A. S. Alghamdi, and A. F. Alharthi, "Hybrid Supervised Machine Learning-based Intrusion Detection System of Internet of Things," *International Journal of Advances in Soft Computing & Its Applications*, vol. 16, no. 2, 2024.

[4] A. Al-Dahoud, M. Fezari, A. A. Alkhatib, M. N. Soltani, and A. Al-Dahoud, "Forest fire detection system based on low-cost wireless sensor network and internet of things," *WSEAS Transactions on Environment and Development*, vol. 19, pp. 506–513, 2023. https://doi.org/10.37394/232015.2023.19.49.

[5] M. Fezari, M. T. Gossa, M. A. Bensaid, and A. Al-Dahoud, "Exploring SBC with Kinect and Machine Learning for Indoor Mobile Robot Design and Monitoring," in *2025 12th International Conference on Information Technology (ICIT)*, pp. 272–276, IEEE, 2025.

[6] R. Singh and S. S. Gill, "Edge AI: a survey," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 71–92, 2023. https://doi.org/10.1016/j.iotcps.2023.02.004.

[7] S. S. Gill, M. Golec, J. Hu, M. Xu, J. Du, H. Wu, G. K. Walia, S. S. Murugesan, B. Ali, M. Kumar, and K. Ye, "Edge AI: A taxonomy, systematic review and future directions," *Cluster Computing*, vol. 28, no. 1, p. 18, 2025. https://doi.org/10.1007/s10586-024-04686-y.

[8] V. Shankar, "Edge AI: a comprehensive survey of technologies, applications, and challenges," in *2024 1st International Conference on Advanced Computing and Emerging Technologies (ACET)*, pp. 1–6, IEEE, 2024. https://doi.org/10.1109/ACET61898.2024.10730112.

[9] T. Meuser, L. Lovén, M. Bhuyan, S. G. Patil, S. Dustdar, A. Aral, S. Bayhan, C. Becker, E. De Lara, A. Y. Ding, and J. Edinger, "Revisiting edge AI: Opportunities and challenges," *IEEE Internet Computing*, vol. 28, no. 4, pp. 49–59, 2024. https://doi.org/10.1109/MIC.2024.3383758.

[10] A. Aldahoud, M. Fezari, A. Al-Dahoud, and D. Aqel, "A Systematic Review on Applying Machine Learning and Deep Learning on SBCs," *WSEAS Transactions on Information Science and Applications*, vol. 21, pp. 272–281, 2024. https://doi.org/10.37394/23209.2024.21.26.

[11] V. N. Pamadi and P. Singh, "Edge AI vs Cloud AI: A Comparative Study of Performance Latency and Scalability," *International Journal of Research in Modern Engineering and Emerging Technology (IJRMEET)*, vol. 13, no. 3, pp. 13–35, 2025.

[12] Y. Shi, K. Yang, Z. Yang, and Y. Zhou, *Mobile Edge Artificial Intelligence: Opportunities and Challenges*. Academic Press, 2021.

[13] S. Mittal, "A survey on optimized implementation of deep learning models on the NVIDIA Jetson platform," *Journal of Systems Architecture*, vol. 97, pp. 428–442, 2019.

[14] T. Sipola, J. Alatalo, T. Kokkonen, and M. Rantonen, "Artificial intelligence in the IoT era: A review of edge AI hardware and software," in *2022 31st Conference of Open Innovations Association (FRUCT)*, pp. 320–331, IEEE, 2022.

[15] G. S. Nikolić, B. R. Dimitrijević, T. R. Nikolić, and M. K. Stojcev, "A survey of three types of processing units: CPU, GPU and TPU," in *2022 57th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST)*, pp. 1–6, IEEE, 2022.

[16] S. Williams, A. Waterman, and D. Patterson, "Roofline: an insightful visual performance model for multicore architectures," *Communications of the ACM*, vol. 52, no. 4, pp. 65–76, 2009.

[17] Y. E. Wang, G.-Y. Wei, and D. Brooks, "Benchmarking TPU, GPU, and CPU Platforms for Deep Learning," *arXiv:1907.10701*, 2019.

[18] V. Sze, Y.-H. Chen, T.-J. Yang, and J. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[19] S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," in *Proc. ICLR*, 2016.

[20] V. Dedeoglu, S. Malik, G. Ramachandran, S. Pal, and R. Jurdak, "Blockchain meets edge-AI for food supply chain traceability and provenance," in *Comprehensive Analytical Chemistry*, vol. 101, pp. 251–275, Elsevier, 2023.

[21] V. K. Rathi, N. K. Rajput, S. Mishra, B. A. Grover, P. Tiwari, A. K. Jaiswal, and M. S. Hossain, "An edge AI-enabled IoT healthcare monitoring system for smart cities," *Computers & Electrical Engineering*, vol. 96, art. 107524, 2021.

[22] M. A. Rahman, M. F. Shahrior, K. Iqbal, and A. A. Abushaiba, "Enabling Intelligent Industrial Automation: A Review of Machine Learning Applications with Digital Twin and Edge AI Integration," *Automation*, vol. 6, no. 3, p. 37, 2025.

[23] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. AISTATS*, pp. 1273–1282, 2017.

[24] C. Frenkel, M. Lefebvre, J.-D. Legat, and D. Bol, "A 0.086-mm² 12.7-pJ/SOP 64k-Synapse 256-Neuron Online-Learning Digital Spiking Neuromorphic Processor in 28-nm CMOS," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 1, pp. 145–158, 2019.

APPENDIX

Table 1. A qualitative comparison of the AI hardware accelerators: VPU, GPU, NPU, and TPU

Aspect	VPU	GPU	NPU	TPU
Purpose	Computer-vision tasks (image/video processing, detection, tracking, segmentation).	Graphics rendering; versatile parallel computing including AI training/inference and simulation.	Accelerating AI/ML inference and neural-network workloads on edge devices.	Google-designed accelerator for ML training and inference, excelling at tensor operations.
Workload	Convolution, feature extraction, motion estimation, depth sensing.	Highly parallel tasks: rendering, deep-learning training, scientific computing, analytics.	Neural-network inference: convolution, pooling, activations, low-precision matmul.	Tensor and linear-algebra operations; large-scale, low-precision computation.
Latency	Low-latency inference for real-time vision.	Varies with workload; throughput is often prioritized over latency in training.	Optimized for low-latency edge inference.	High throughput and low latency for large-scale inference.
Efficiency (perf/W)	Very high for vision-specific tasks; minimal power.	Moderate for AI inference; high peak for training; significant power.	Very high for edge inference; low-power operation.	High for tensor workloads; strong perf/W in newer generations.
Power	Low to very low; suited to embedded/battery systems.	Moderate to very high; high-end needs cooling.	Low to moderate; mobile/edge optimized.	Generally lower than high-end GPUs for similar inference.
Software ecosystem	Vendor-specific (e.g., Intel OpenVINO); growing vision support.	Mature: CUDA, ROCm, OpenCL; TensorFlow, PyTorch, JAX.	Growing: TensorFlow Lite, ONNX, embedded SDKs.	Primarily TensorFlow with JAX/PyTorch via XLA.
Scalability	Limited; single-device or small-scale.	Highly scalable (multi-GPU, clusters).	Via SoC integration or edge platforms; cloud NPUs are emerging.	Massive cloud scalability (TPU Pods).
Applications	Autonomous vehicles, surveillance, robotics, medical imaging, AR/VR, drones.	Gaming, content creation, simulation, analytics, cloud AI training.	Mobile/IoT devices, smart-home, robotics, wearables.	Google AI services, recommendation, large language models, cloud AI.
Cost	Cost-effective in volume for embedded vision.	Wide range, up to expensive data-center parts.	Lower than high-end GPUs; cost-efficient in SoCs.	Higher hourly cloud cost; cost-effective at scale; low-cost Coral USB for edge.

Source: created by the authors.

Table 2. Representative quantitative benchmark of edge AI accelerator classes. Figures are public peak values (INT8 unless noted) and are workload-dependent; they should be re-verified against current datasheets and MLPerf-audited results

Metric	VPU (Intel Movidius Myriad X / NCS2)	GPU (NVIDIA Jetson AGX Orin)	NPU (Hailo-8 / mobile-SoC NPU)	TPU (Google Coral Edge TPU)
Peak throughput (INT8)	≈ 1–4 TOPS	up to ≈ 275 TOPS	≈ 26 TOPS (Hailo-8)	4 TOPS
Typical power	≈ 1–2 W	15–60 W	≈ 2.5 W	≈ 2 W
Efficiency (TOPS · W^{-1})	≈ 1–2	≈ 4.6–8	≈ 10	2
On-chip memory	small (SHAVE-local)	shared LPDDR5	on-chip SRAM	8 MB
Numeric precision	INT8 / FP16	INT8 / FP16 / FP32	INT8 / INT4	INT8 only
Representative result	object detection on a USB stick	runs all six MLPerf Edge tests	real-time YOLO	MobileNetV2 ≈ 400 fps (≈ 2–3 ms)
Best-fit role	embedded vision	server-class edge/robotics	high-efficiency vision IoT	quantized TFLite inference

Source: compiled by the authors from [12], [14], [15].

Table 3. Positioning of this review against representative recent edge AI surveys

Survey	Hardware accelerators	Formal equations	Quantitative benchmarks	Placement algorithm	Applications
Singh & Gill (2023) [6]	Partial	No	Partial	No	Yes
Gill et al. (2025) [7]	Partial	No	Partial	No	Yes
Shankar (2024) [9=8]	Partial	No	No	No	Yes
Meuser et al. (2024) [10=9]	Partial	No	No	Partial	Yes
This review	**Yes (VPU/GPU/NPU/TPU)**	**Yes (Eqs. 1–26)**	**Yes (Table 2)**	**Yes (Algorithm 1)**	**Yes**

Source: created by the authors

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.