

Article

Understanding Energy Efficiency of AI Deployments in IoT-Driven Smart Cities

Salvatore Bramante ¹, Filippo Ferrandino ² and Alessandro Cilardo ^{2,*}¹ Scuola IMT Alti Studi Lucca, 55100 Lucca, Italy; salvatore.bramante@imtlucca.it² Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione, Università degli Studi di Napoli Federico II, 80125 Napoli, Italy; filippo.ferrandino@unina.it

* Correspondence: acilardo@unina.it

Abstract

The pervasive adoption of AI and AIoT applications at the network edge presents both opportunities and challenges for smart cities. With a focus on the energy efficiency of AI in urban environments, this paper provides a systematic comparative analysis of representative edge hardware platforms, i.e., embedded GPUs, FPGAs, and ultra-low-power microcontroller-/sensor-class devices, assessing their suitability for AI workloads in IoT-driven smart city infrastructures. The evaluation, based on direct characterization of diverse neural networks and relevant datasets, quantifies computational performance and energy behavior through inference latency, throughput, and energy/per inference measurements. Across the evaluated network-board pairs, the measured inference power spans several orders of magnitude, ranging from 0.1–10 mW for ultra-low-power Intelligent Sensor Processing Units (ISPU) up to 1–10 W for embedded GPUs, highlighting the wide design space between the least and most power-demanding configurations. Results indicate that embedded GPUs provide a favorable performance-to-power ratio for computationally intensive workloads, while MCU/ISPU-class solutions, despite throughput limitations, offer compelling advantages in ultra-low-power scenarios when combined with quantization and pruning, making them well-suited for distributed sensing and actuation typical of smart city deployments. Overall, this comparative analysis guides hardware selection for heterogeneous, sustainable AI-enabled urban services.

Keywords: AI-powered smart cities; edge AI; resource-efficient AI; Artificial Intelligence of Things (AIoT)



Academic Editors: Hakilo Sabit and Peter Han Joo Chong

Received: 25 January 2026

Revised: 3 March 2026

Accepted: 6 March 2026

Published: 8 March 2026

Copyright: © 2026 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 \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

The rapid convergence of Artificial Intelligence (AI) technologies with an increasingly heterogeneous ecosystem of computing architectures is enabling new classes of sustainable smart city solutions [1–4]. As modern Deep Neural Networks (DNNs) continue to grow in complexity and scale, their widespread integration into urban services has magnified the need for hardware platforms explicitly engineered to sustain intensive AI workloads with high efficiency [5]. These workloads now underpin key smart city domains such as infrastructure supervision, industrial automation, intelligent mobility systems, environmental sensing, and the expanding Internet of Things (IoT) fabric that fuels city-wide digital services. In this context, selecting the appropriate hardware platform for Machine Learning (ML) tasks becomes a decisive factor that directly influences the feasibility, responsiveness, and energy footprint of smart city deployments.

To address all the above challenges, three foundational hardware paradigms have emerged for AI acceleration, GPUs [6], FPGAs [7–9], and MCUs [10,11], each characterized by distinctive architectural principles and energy–performance trade-offs.

At the ultra-low-power end, this includes Intelligent Sensor Processing Units (ISPU), i.e., sensor-class devices that enable near-sensor inference within mW budgets. Overall, the appropriate placement of a ML task across these tiers determines the achievable latency–energy trade-off and the feasibility of large-scale urban deployments.

Through a detailed evaluation of energy consumption per frame, inference latency, and throughput across representative workloads, it is illustrated how each platform class aligns with the operational demands of modern IoT-driven smart cities. Comparative analysis offers practical guidance for designing next-generation AI infrastructures that meet both performance requirements and sustainability objectives in urban ecosystems. Despite extensive work on individual tiers, prior smart-city studies rarely provide a unified evaluation that simultaneously (i) spans MCU, ISPU, FPGA, and GPU targets; (ii) uses the same task/dataset across platforms; and (iii) reports comparable device-level energy and performance metrics. The main contributions of this study are as follows:

- **Comprehensive Energy Profiling for Sustainable Urban AI Deployments:** A rigorous analysis of the power consumption patterns associated with executing diverse machine learning models on intelligent sensor platforms under representative operating conditions is provided, supporting the design of energy-efficient IoT-driven smart city architectures.
- **Cross-Platform Evaluation Framework for Artificial Intelligence of Things (AIoT) Services in Smart Cities:** A decision-oriented comparative analysis, mapping heterogeneous embedded platforms to workloads of varying complexity, is introduced. This framework integrates key performance metrics, energy efficiency indicators, and application-specific suitability factors, delivering practical guidelines for informed hardware selection in power-sensitive smart city deployments.
- **An ISPU-based Dataset for Gesture Recognition:** ISPU_Dat is introduced and publicly released, a curated gesture-recognition dataset optimized for resource-constrained embedded systems. This dataset supports reproducible research and benchmarking in ultra-low-power embedded AIoT, enabling the development of sustainable smart city services that rely on efficient edge intelligence.

2. Related Work

Power and energy for AI inference are increasingly studied as models are increasingly deployed on resource-constrained devices, spanning methodological work on how to measure energy and how efficiency trends reflect both hardware specialization and algorithmic improvements [12–14]. At the platform level, benchmarking studies compare heterogeneous accelerators and deployment metrics across edge systems (including the impact of quantization and even standby power) [15–17]. Within the FPGA-centric literature, several works analyze domain-specific deployments or compare FPGAs against GPUs across workloads, highlighting energy–latency trade-offs but largely concentrating on mid-to high-power platforms [18–27]. Complementary research targets TinyML and on-sensor intelligence as well as smart-city application stacks and architectural surveys, typically focusing on a single device family or a single service rather than cross-tier comparisons on a common protocol [28–54].

Although the existing literature offers rich insights into energy measurement and smart city applications, it remains fragmented. This work offers a broad view of smart-city-relevant domains (Figure 1) and multiple hardware platforms under a unified energy-profiling methodology.

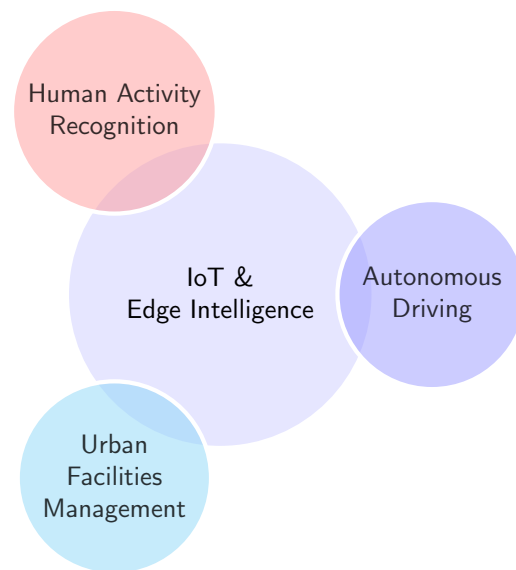


Figure 1. Main smart city application domains considered in this work.

To make the comparison across a heterogeneous literature explicit, we adopt five tags that we use throughout this section and in Table 1: (i) SC (smart-city relevance): the work is explicitly framed around smart-city use cases or evaluated in an urban domain; (ii) Tiny (TinyML): the focus is on microcontroller- or sensor-class deployment; (iii) HW-div (hardware diversity): results include a comparison across multiple hardware classes (e.g., MCU/ISPU vs. FPGA vs. embedded GPU) rather than a single device family; (iv) Energy (energy measurement): the work provides direct energy/power measurements; (v) Data/Tool: the work releases tooling/benchmarks, enabling reproducible energy profiling. These tags are not meant to rank contributions, but to clarify the coverage of the design space each work addresses.

Representative contributions are summarized in Table 1 and organized by domain and focus in the following cross-domain analysis:

1. Some studies focus on concrete smart city scenarios but typically remain confined to one application and one or two hardware families. Most mobility- and traffic-oriented studies [19–22,43–45,47–50,55,56] only use a single class of accelerator (typically a Field Programmable Gate Array (FPGA) or embedded Graphics Processing Unit (GPU)). Similarly, works on urban services and environment [26,36,38–40] concentrate on specific applications (environmental monitoring, waste management, smart grid) with limited device-level energy analysis. In terms of Table 1, these studies usually score positively on the SC column but rarely on *Tiny* or *HW-div*, as well as on their energy evaluations.
2. Other works [28–35] focus on TinyML and on-sensor processing. These contributions explore low-resource constraints, developing specialized models for Micro Controller Units (MCUs) and intelligent sensors, and in some cases, address smart city use cases. However, they generally consider a single device family (one MCU or one ISPU), do not include custom or consumer accelerators (FPGAs, GPUs) and rarely provide a public dataset dedicated to sensor-level gesture interaction in urban contexts. In Table 1, these works populate the *Tiny* column but leave *HW-div* and SC often empty. By contrast, this study benchmarks all platforms side by side and introduces the ISPU_Dat dataset for embedded gesture recognition in smart city scenarios.
3. Hardware-centric and algorithmic benchmarks that do not explicitly target urban applications [10,23,24,27,57]. These works typically score well on *Energy* and sometimes on *HW-div*, but they are not framed around smart city workloads or sensor-to-cloud

hierarchies. This contribution assesses standard models and toolchains against urban tasks and edge constraints.

- Another set of contributions operates at the architectural or conceptual level [46,51–54], discussing edge AI, cloud–edge orchestration, and energy-aware scheduling in smart cities and smart grids. They provide valuable system-level perspectives and score positively in the *Data/Tool* or *SC* columns, but they generally treat the underlying hardware as an abstract resource with fixed power–performance characteristics. In contrast, this work provides directly measured energy consumption and latency across a wide range of realistic workloads on heterogeneous platforms, thereby establishing a quantitative low-level foundation upon which future edge–cloud frameworks can develop more accurate schedulers and energy models.

Table 1. For each study, papers are grouped into macro-categories and a representative example is reported. The table uses checkmarks to indicate whether a study (*SC*) is framed in a smart-city or closely related urban scenario; (*Tiny*) targets MCU-class or on-sensor TinyML devices; (*HW-div*) performs a cross-platform hardware comparison beyond a single device family; (*Energy*) reports device-level power/energy measurements or models; and (*Data/Tool*) introduces a dataset, benchmark, or reusable framework. The last column highlights the main differences with respect to this work.

Category	Analyzed Works	Representative Study	SC	Tiny	HW-Div	Energy	Data/Tool	Main Difference vs. This Work
Urban services and environment	[26,36,38–40,42–49]	[36]: IoT-based environmental monitoring using low-cost sensors and simple computation nodes.	✓	–	–	–	–	Application-focused IoT monitoring without ML acceleration and without device-level or multi-tier energy benchmarking.
TinyML/on-sensor processing	[24,28–35]	[32]: On-sensor gesture recognition using LSM6DSV16X (ISPU) with decision-tree inference in automotive and driving-monitoring contexts.	–	✓	–	–	–	Demonstrates true on-sensor inference but does not compare against GPU/FPGA tiers nor provide an open smart-city gesture dataset.
Hardware benchmarks	[10,19–25,27,55,57]	[27]: FPGA–GPU comparison for AI workloads, measuring throughput, flexibility, latency and power consumption.	–	–	✓	✓	–	Strong HW/energy analysis but not contextualized to smart-city tasks, and no sensor-to-cloud tier comparison.
Architectural frameworks	[46,51–54]	[52]: Survey on edge AI, blockchain and distributed architectures for smart-city applications, focusing on governance and interoperability.	✓	–	–	–	✓	High-level architectural analysis treating hardware as abstract resource; no device-level energy measurements nor cross-tier profiling.

The resulting picture is not tied to a single application or device family, but instead offers a continuous view of energy and latency that can support informed hardware selection for sustainable AI deployments in smart cities.

This study contributes to the TinyML literature in two ways. First, it conducts a unified measurement campaign that directly compares ISPU platforms, MCU-class devices and MCU + NPU systems on-sensor. Secondly, it releases ISPU_Dat, a gesture-recognition dataset designed for resource-constrained embedded sensing that is used consistently across platforms. Together, these contributions allow for realistic, quantitative comparisons of ISPU-class solutions with conventional MCUs and higher-end accelerators in smart city-relevant tasks.

3. Methodology

This study takes a systematic approach to evaluating energy efficiency across a range of hardware platforms executing various AIoT workloads in smart cities. This methodology

is designed to ensure accurate power and performance measurements, fair cross-platform comparisons, consistent results and thorough coverage of typical edge AI applications.

3.1. Overview

A comprehensive set of hardware platform classes, representing the typical categories of edge AI systems that are relevant to smart city applications, is evaluated. These range from ultra-low-power microcontrollers to high-performance embedded GPUs:

- Ultra-low-power sensors with integrated AI acceleration (ISPU-class);
- Mid-range MCU-class devices with a dedicated NPU;
- High-end MCU-class devices without a dedicated AI accelerator;
- Embedded GPU (eGPU) platforms;
- FPGA-based heterogeneous platforms.

The specific platforms used for the experiments are detailed in Section 3.6. This section showcases the entire range of edge AI deployment options, from sensor-level processing to powerful edge appliances used in urban settings. This enables a thorough analysis of the trade-offs between power and performance across a variety of computational capabilities.

To represent the diverse requirements of real-world smart city applications that rely on AIoT, classes of models are identified that span various levels of computational complexity. As illustrated in Figure 1, each workload is mapped to one of four smart city domains:

- Human Activity Recognition (HAR): This encompasses AI workloads for human-machine interaction and behavioral monitoring in urban environments, such as gesture recognition and activity classification. These tasks are typically addressed using traditional machine learning techniques and lightweight convolutional neural networks, and are well-suited to execution on sensors and ultra-low-power MCUs.
- IoT and Edge Intelligence: This represents generic classification and inference tasks commonly deployed on distributed IoT nodes, including local event detection and simple decision making at the edge. This domain is characterized by compact fully connected or shallow neural network models designed for low-latency and energy-efficient execution.
- Urban Facility Management: This covers AI-driven analysis for monitoring and managing urban infrastructure, including asset inspection, land-use analysis, and infrastructure condition assessment. These applications rely on deeper convolutional architectures for image classification and semantic segmentation, with moderate to high computational demands.
- Autonomous Driving and Urban Mobility: This includes perception-centric workloads supporting intelligent transportation systems, such as object detection, scene understanding, and traffic analysis. These tasks are typically implemented using state-of-the-art deep learning models with high computational intensity, stressing high-performance embedded platforms.

To ensure a meaningful comparison, all workloads are structured as a three-stage pipeline: pre-processing, inference, and post-processing. This model captures not only the neural network cost but also the energy overhead of data preparation (e.g., resizing, windowing) and decision logic (e.g., NMS for YOLO, voting for HAR). While high-performance platforms are dominated by core inference, on constrained MCUs and sensors, pre- and post-processing can consume a non-negligible fraction of the total energy. Unlike related works (Table 1) that often focus on single hardware tiers, this unified methodology allows us to place sensor-level TinyML and high-throughput perception on a common energy-time plane.

Production-grade optimizations are applied to ensure fair comparisons: INT8 quantization for MCUs and NPUs; platform-specific compilation (TensorRT for NVIDIA, STM32Cube.AI for STM32, DPU compiler for Xilinx); utilization of hardware accelerators; and memory-efficient buffer management. These settings reflect realistic deployment scenarios rather than naive implementations.

To evaluate the power consumption and inference performance of AI models across different embedded platforms, distinct datasets are selected representing specific ML tasks. The selection ensures realistic benchmarks ranging from computer vision to sensor-based gesture recognition.

- **Aeroscapes [58]:** This is selected for semantic segmentation tasks using the U-Net architecture. This dataset consists of aerial imagery captured from drones (5–50 m altitude) and provides a unique bird’s-eye view of urban environments. It includes annotated classes such as vegetation, roads, and vehicles, with a realistic class imbalance suitable for autonomous navigation scenarios.
- **Microsoft COCO [59]:** This is utilized for object detection using the You Only Look Once (YOLO) architecture. Containing 80 object categories in complex, natural environments, COCO serves as a rigorous benchmark for evaluating real-time processing capabilities and accuracy on embedded hardware.
- **ImageNet [60]:** Used for image classification benchmarks, ImageNet contains over one million labeled images across 1000 object categories, and is widely adopted to assess classification accuracy and performance trade-offs on embedded platforms.
- **ISPU_Dat [61]:** This is a custom dataset created specifically for gesture recognition tasks on Intelligent Sensor Processing Unit (ISPU) sensors. Data was collected using MEMS Studio [62] on an STM32 Nucleo-64 board. It comprises 583 measurements of three gestures (phone call, wake-up, hello) sampled at 26 Hz. Each sample is a 6-dimensional tuple (3-axis gyroscope and accelerometer), designed to maintain the low computational footprint required for sensor–edge inference.

Unless otherwise stated, an inference refers to one end-to-end pipeline execution (covering pre-processing, model execution, and post-processing) for a single input sample.

3.2. Detail of the AI Models

A diverse set of models for MLs and DNNs was selected to evaluate inference performance and energy consumption across different hardware platforms. The workloads span lightweight structured-data methods commonly found in smart city sensing pipelines to computationally intensive computer vision algorithms, enabling a comprehensive assessment of processing capabilities, acceleration opportunities, and power-efficiency limits.

The quantitative results in terms of INT8 classification and detection accuracy for each model–dataset pair are summarized in Table 2. For structured-data processing, this study focused on gesture recognition, employing Decision Trees and Random Forests as representative models. Decision Trees offer a deterministic, lightweight baseline well-suited for evaluating low-latency, ultra-low-power behavior on embedded systems. Random Forests, while more computationally demanding, introduce ensemble-based robustness and moderate parallelism, providing a more challenging yet realistic test for platforms such as the STM32N6570-DK.

In the context of visual understanding, the analysis incorporated classification workloads using a progression of neural architectures. Dense Neural Networks served as an intermediate reference bridging traditional ML and deep learning. Convolutional Neural Networks (CNNs) [63] were then used to examine each platform’s ability to handle matrix- and convolution-intensive operations. Among them, VGG16 [64] was selected as

a standardized benchmark due to its deep, uniform structure, which effectively stresses computational throughput and energy efficiency on embedded hardware.

Table 2. Accuracy of the evaluated AI models on their reference datasets. All neural network models are reported after post-training INT8 quantization, whereas Decision Tree and Random Forest results are reported for their native implementations.

AI Model	Input Size	Dataset	Accuracy (%)
Decision Tree	6×26	ISPU_Dat	94.5
Random Forest	6×26	ISPU_Dat	96.5
LeNet	6×26	ISPU_Dat	99.1
Dense (MLP)	6×26	ISPU_Dat	98.8
MobileNetv2	224×224	ImageNet	71.3
ResNet50	224×224	ImageNet	75.9
VGG16	224×224	ImageNet	70.9
YOLOv8	256×256	COCO	37.5
YOLOv11	640×640	COCO	52.0
U-Net	256×256	Aerospaces	83.5

To evaluate high-complexity inference pipelines, this study included object detection tasks based on the YOLO architecture, specifically versions v8 and v11. Unlike standard classifiers, YOLO reformulates detection as a single regression problem, resulting in significantly higher computational and memory requirements. These properties make it an ideal candidate for measuring the benefits of hardware acceleration, such as GPU and Neural Processing Unit (NPU) support on NVIDIA Jetson platforms, while simultaneously exposing the challenges faced by resource-constrained microcontrollers.

Finally, the workload suite incorporated semantic segmentation through the U-Net architecture [65]. Characterized by its symmetric encoder–decoder topology and extensive use of skip connections, U-Net effectively combines fine-grained spatial features with high-level semantic information. Its fully convolutional structure supports flexible input resolutions, making it a meaningful test case for assessing memory throughput, buffer management, and execution efficiency on a wide range of embedded platforms.

3.3. Execution Framework and Analysis

To ensure consistent evaluations, a standardized workload execution framework featuring a custom automated harness was developed. This system coordinates input submission, result capture, and power measurement across controlled environments. The subsequent analysis focuses on extracting actionable insights through four key steps:

1. Offset Power Isolation: Separating baseline platform power from dynamic power to analyze computational efficiency independent of fixed overheads.
2. Energy Metrics: Calculating total energy per inference to provide standardized cross-platform comparisons.
3. Efficiency Frontier Analysis: Plotting energy consumption against inference time to identify Pareto-optimal configurations representing the best trade-offs.
4. Scaling Trends: Examining how energy scaling correlates with computational complexity to reveal architectural patterns.

This integrated approach allows for a rigorous assessment of the power–performance trade-offs essential for constrained edge AI systems.

To decouple the energy cost of the AI workload from the baseline consumption of the platform, we apply offset-power isolation. We define the idle (offset) power P_{idle} as the average electrical power drawn by the Device Under Test when it is powered on and in the same operational state used for the experiment (same peripherals enabled, same

OS/firmware configuration, same power mode), but without executing any AI workload. In practice, P_{idle} is measured over a time window T_{idle} before (or after) each run, using the same runtime configuration and binary as the corresponding workload. As a result, P_{idle} can slightly vary across model implementations on the same device (e.g., due to different libraries, memory allocations, or enabled peripherals).

3.4. Energy Efficiency Metrics

To facilitate meaningful cross-platform comparisons, we report complementary energy–performance metrics that capture both end-to-end deployment cost and workload-attributable cost.

- Energy per inference (end-to-end), E_{tot} [J/inference]: This is the total energy consumed to execute one full pipeline run (pre-processing + model execution + post-processing) for a single input sample, including baseline (idle) power.
- Energy per inference (active only), E_{act} [J/inference]: This is the workload-attributable energy obtained via offset-power isolation, i.e., the integral of $(P(t) - P_{\text{idle}})$ over the execution window.
- Energy–Delay Product, EDP [J·s]: This is defined as $\text{EDP} = E_{\text{tot}} \cdot T$, where T is the end-to-end latency of one pipeline run. EDP captures the combined energy and latency cost and is useful to rank configurations when neither energy nor latency alone is sufficient.

EDP, together with E_{tot} and E_{act} , provides a clearer and more decision-oriented view of the trade-offs relevant to smart-city AIoT deployments.

3.5. Measurement Infrastructure

A measurement setup was implemented to obtain power measurements (Figure 2).

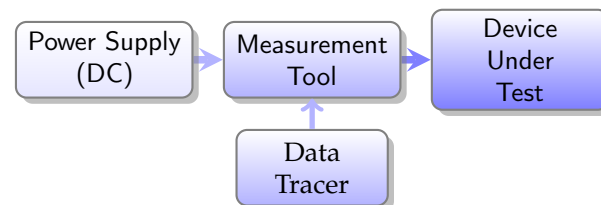


Figure 2. Schematic representation of the power measurement setup, showing the Device Under Test (DUT), precise external power measurement instruments, and data acquisition system.

For each DUT, the following measurement protocols are established:

- External high-precision instrumentation: Calibrated digital multimeters are employed with μA precision and sampling rates of at least 1 kHz to capture dynamic power variations. These instruments were positioned between the power supply and DUT to measure the total power of the system.
- Onboard telemetry: Where available, platform-specific power monitoring capabilities (e.g., NVIDIA’s *jtop* for NVIDIA Jetson platforms, PYNQ power APIs for AMD Zynq UltraScale+ MPSoC ZCU102) are utilized to provide complementary measurements and component-level power breakdowns.
- Temporal correlation: All power measurements were precisely time-synchronized with workload execution using trigger signals to ensure accurate correlation between computational phases and power consumption patterns.
- Thermal stabilization: Each platform was allowed to reach thermal equilibrium before measurement to eliminate warm-up effects and ensure consistent power readings.

- Statistical rigor: Each measurement was repeated at least 30 times, to ensure statistical significance and reproducibility.

To capture the complete performance profile of each platform–model combination, the following metrics were measured:

- Inference latency: This is the time from input submission to result availability, measured at microsecond resolution.
- Throughput: This is the number of inferences per second under sustained load.
- Model accuracy: This is to ensure optimizations do not compromise functional requirements.
- Memory utilization: This is the peak and average memory consumption during inference.

Hardware performance counters and platform-specific profiling tools were used to analyze subsystem activity during inference on platforms with complex System on Chip (SoC) architectures.

3.6. Analyzed Platforms

Diverse hardware platforms were selected to cover a broad spectrum of computing paradigms, ranging from ultra-low-power sensor processing to high-performance edge computing. At the lowest energy tier, the analysis focused on Intelligent Sensor Processing Units, using the *LSM6DSO16IS* (mounted on the *X-NUCLEO-IKS4A1*). This inertial sensor integrates an ISPU, a programmable core optimized for executing AI algorithms directly in the digital domain. The ISPU includes separate volatile memory banks for data (8 KB) and program (32 KB), both loaded at boot via SPI. By processing signals internally with its 32-bit core before they exit the sensor, it reduces data transfer latency and system-level energy consumption, making it suitable for always-on battery-powered applications.

To assess model inference on microcontrollers, two complementary approaches were examined. The *STM32N6570-DK* represents a next-generation design, embedding an Arm Cortex-M33 complemented by a dedicated NPU that accelerates quantized 8-bit integer (INT8) models (converted using STM32Cube.AI). This hardware offloading significantly improves energy efficiency for CNN workloads. In contrast, the *STM32H747I-DISCO* served as a high-end general-purpose baseline: in addition to a Cortex-M4 core, its heterogeneous dual-core architecture relies on a Cortex-M7 microcontroller unit to run inference without specialized acceleration, highlighting the performance gap relative to NPU-enabled MCUs.

For hardware-accelerated computation, this study employed the Field-Programmable Gate Array platform *Zynq UltraScale+ MPSoC ZCU102*. This heterogeneous system combines a Processing System (Quad-core Cortex-A53 APU, Cortex-R5 RPU, and Mali-400 GPU) with Programmable Logic (PL). Deep learning workloads were offloaded to the Xilinx Deep Learning Processor Unit (DPU), a soft-core IP instantiated in the PL that executes neural networks (e.g., VGG, YOLO) through a pipelined architecture. The DPU fetches instructions and data from shared memory and coordinates with the APU via interrupts. In addition to the ZCU102 evaluation board, we also considered the *AMD Kria KV260*, which integrates a Zynq UltraScale+ MPSoC System-on-Module (SOM) and a carrier designed for vision workloads, offering a practical form factor and a vendor-supported software stack for DPU-based acceleration.

Finally, high-throughput workloads were benchmarked on Single-Board Computers from the NVIDIA Jetson family. The *NVIDIA Jetson Nano* provides an entry-level platform using a 128-core Maxwell GPU to accelerate FP16 and INT8 inference via TensorRT, though it lacks a dedicated NPU and exhibits moderate idle power consumption. At the high end, the *NVIDIA Jetson AGX Orin* offers server-class performance, to 275 Trillions of Operations Per Second (TOPS), through an Ampere GPU and dedicated Deep Learning Accelerator (DLA), supporting computationally intensive pipelines such as 3D perception

and advanced object detection (e.g., YOLOv8/v11). Its substantial processing capabilities come with higher energy demands (15–60 W), positioning it for grid-powered industrial and smart infrastructure deployments. TensorRT versions used in our setup were 8.0 for NVIDIA Jetson Nano devices on the JetPack 4.6 stack and 8.5 on Orin platforms on the JetPack 5.

For platforms equipped with active cooling (AMD Zynq UltraScale+ MPSoC ZCU102, AMD Kria KV260, NVIDIA Jetson Nano, NVIDIA Jetson AGX Orin), we kept the default vendor cooling configuration enabled (i.e., stock fan and default fan control policy). No custom fan curves or forced fan speeds were applied. For the remaining boards (MCU-class and sensor-class devices), no active or passive cooling solutions are provided by design, and during our experiments, the devices remained within their normal operating temperature range; therefore, no additional thermal mitigation was required. The devices were operated at their default clock settings. The STM32N6570-DK and STM32H747I-DISCO ran at 800 MHz and 480 MHz, respectively, while the ISPU module was clocked at 10 MHz. On eGPU-based platforms, the NVIDIA Jetson Nano was configured for up to 921 MHz on the GPU and 1.43 GHz on the CPU, and the NVIDIA Jetson AGX Orin for up to 918 MHz on the GPU and 2.0 GHz on the CPU. For the AMD Zynq UltraScale+ MPSoC ZCU102, the DPU frequency was set to 300 MHz.

None of the devices was overclocked; all platforms were tested under standard vendor clock settings. In particular, for the NVIDIA Jetson platforms, we selected the *MaxN* power mode and used the standard performance configuration for that mode (no manual clock offsets).

3.7. Limitations

This study provides a measurement-based comparison across heterogeneous edge platforms; however, several limitations and scope constraints must be considered when interpreting the results and transferring them to other deployments.

Results depend on the chosen datasets and tasks (ImageNet, COCO, Aerscapes, and ISPU_Dat), which were selected to represent recurring smart-city domains while remaining executable on highly resource-constrained devices. As a consequence, the measurements may not capture all data distributions and operational conditions encountered in real urban deployments (e.g., different sensor modalities, lighting/weather regimes, or domain shifts), and the reported trends should be interpreted as representative of the evaluated workload classes rather than universal.

The evaluation benchmarks a fixed set of representative model architectures (e.g., MobileNetV2/ResNet50/VGG16, YOLO variants, U-Net, and lightweight HAR models) in deployment-ready configurations. While quantization and model compression are considered, the study does not perform an exhaustive exploration of alternative architectures, backbones, or neural architecture search; therefore, different design choices (e.g., lighter/heavier backbones, different operator mixes, or altered input resolutions) may shift the corresponding operating points in the energy–latency space.

The reported measurements focus on inference (end-to-end pipeline execution on the device) and do not include the energy cost of model training, fine-tuning, or repeated retraining cycles. Training energy can be significant for large models and may affect lifecycle sustainability considerations; however, it is orthogonal to the core goal of this work, which is to characterize deployment-time energy and latency on edge hardware for smart-city AIoT services.

Cross-platform comparisons necessarily involve different software stacks (e.g., TensorRT, STM32Cube.AI, FPGA DPU toolchains) and runtime libraries. Although production-grade optimizations are applied on each platform, differences in kernels, memory management, and operator support may influence absolute numbers. The goal is to reflect realistic deployment practice rather than isolate idealized accelerator-only performance.

Measurements were performed under vendor-default cooling and clock configurations and repeated to reduce variance. Nevertheless, long-running deployments may exhibit different behavior due to sustained thermal dynamics, DVFS policies, and background services (especially on OS-based platforms). Reported figures represent end-to-end device measurements that include platform baseline power (OS, memory subsystem, and enabled peripherals). Moreover, networking/communication energy in edge–cloud pipelines is outside the measurement boundary; deployments where transmission dominates energy may shift the optimal design point toward more local processing.

4. Results

This section summarizes the main results obtained with the methodology described in Section 3.1. Rather than enumerating all measured configurations, we highlight recurring energy–latency patterns that are consistent across workloads and platform classes. These findings establish the key architectural trade-offs for AIoT deployments in smart-city environments and directly inform the design guidelines in Section 5. A comparative analysis of performance and energy characteristics across various AI workloads for each selected platform is presented. Figures 3–5 illustrate key relationships between power consumption, energy efficiency, and inference time on the evaluated platforms.

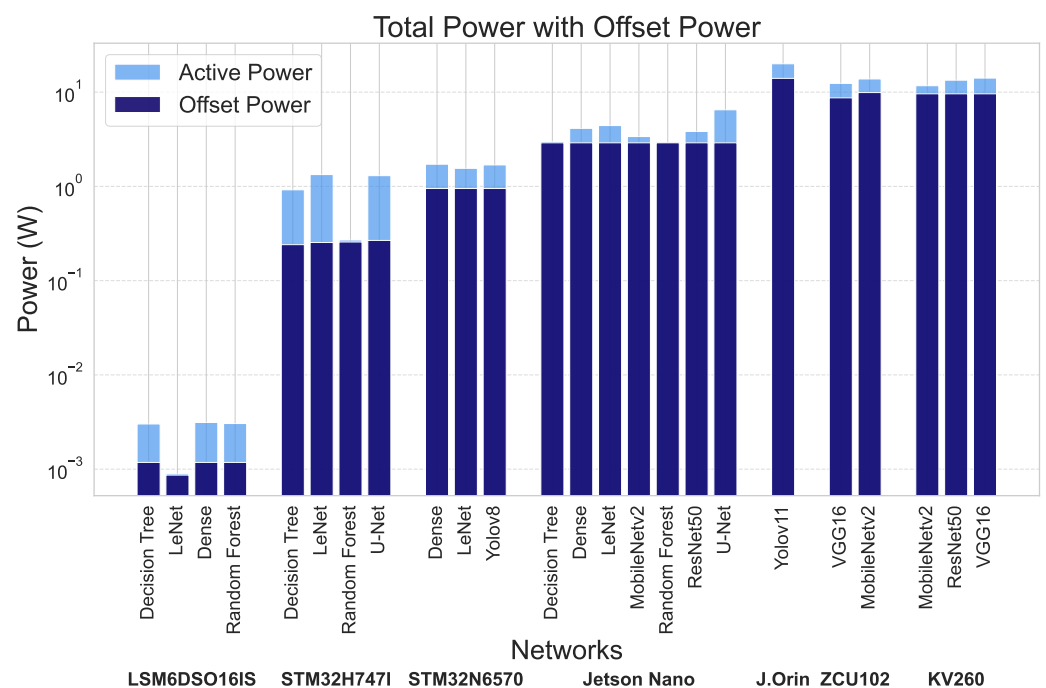


Figure 3. This chart displays neural networks' baseline (offset) power draw and active power consumption during execution. The observed flattening is due to the logarithmic scale.

The distribution of power consumption illustrated in Figure 3 reveals significant variations between hardware platforms. ISPU-class measurements fall in the sub-*mW* to few-*mW* range, making ISPU devices ideal for extremely power-constrained applications. The STM32N6570-DK and STM32H747I-DISCO platforms occupy the middle ground (≈ 0.1 – 1 W), whereas the NVIDIA Jetson family devices and the AMD Zynq UltraScale+

MPSoC ZCU102 consume significantly more power ($\approx 1\text{--}10\text{ W}$). In particular, baseline offset power often dominates total consumption, particularly in higher-performance platforms, indicating that system-level power management is as essential as model optimization for energy efficiency.

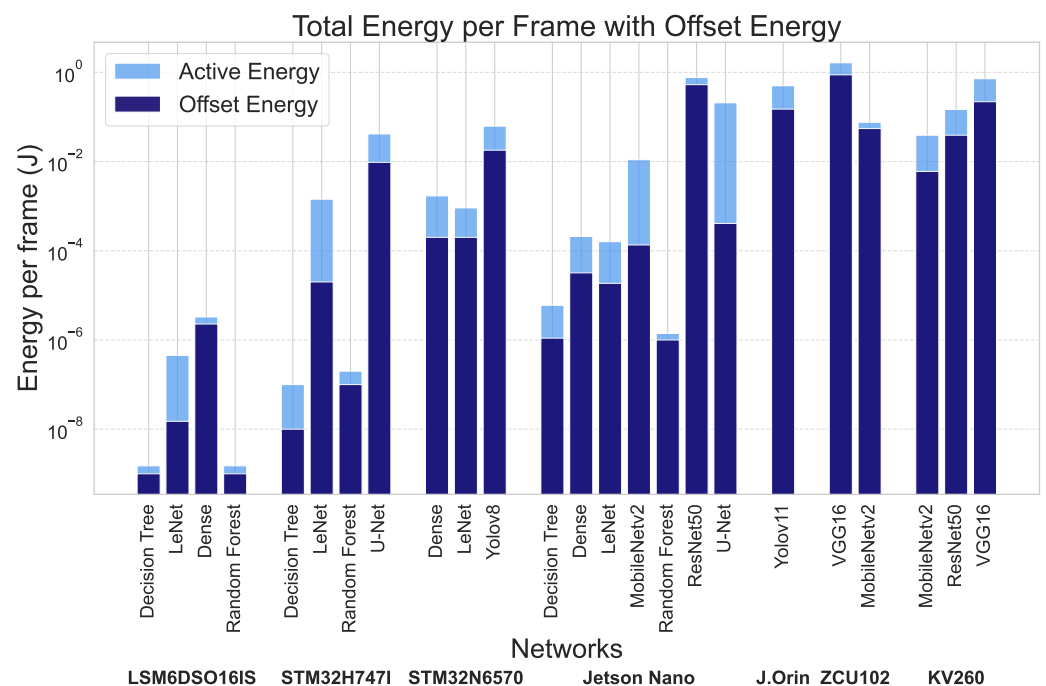


Figure 4. Breakdown of energy per inference into baseline (idle) and active components. For platforms with higher idle power, the baseline term can dominate short workloads; for slower workloads, the idle contribution grows proportionally with execution time.

Figure 4 extends the analysis to the energy consumption per inference frame, which is the product of power consumption and inference time. This metric provides insight into the total energy cost of processing a single input, which is particularly relevant for battery-operated devices. The stark differences between platforms, spanning nine orders of magnitude from nanojoules to joules, demonstrate how hardware selection significantly affects energy efficiency. Although more powerful platforms consume more absolute energy, the relationship between computational capability and energy consumption is not linear across different hardware architectures, highlighting the importance of platform-specific optimization strategies.

Figure 5 provides a holistic view of the performance–energy trade-off across all the platforms and models evaluated. The apparent correlation between inference time and energy consumption within each platform family establishes distinct performance clusters: ultra-low-power devices (bottom left), microcontroller-class devices (middle), and high-performance platforms (top right). This visualization illustrates the efficient frontier system designers must navigate when selecting hardware platforms for AI applications with specific latency and power constraints. Table 3 reports the mean end-to-end power, latency, energy per inference, and EDP for each network–board pair; these values correspond to the points shown in Figures 3–5.

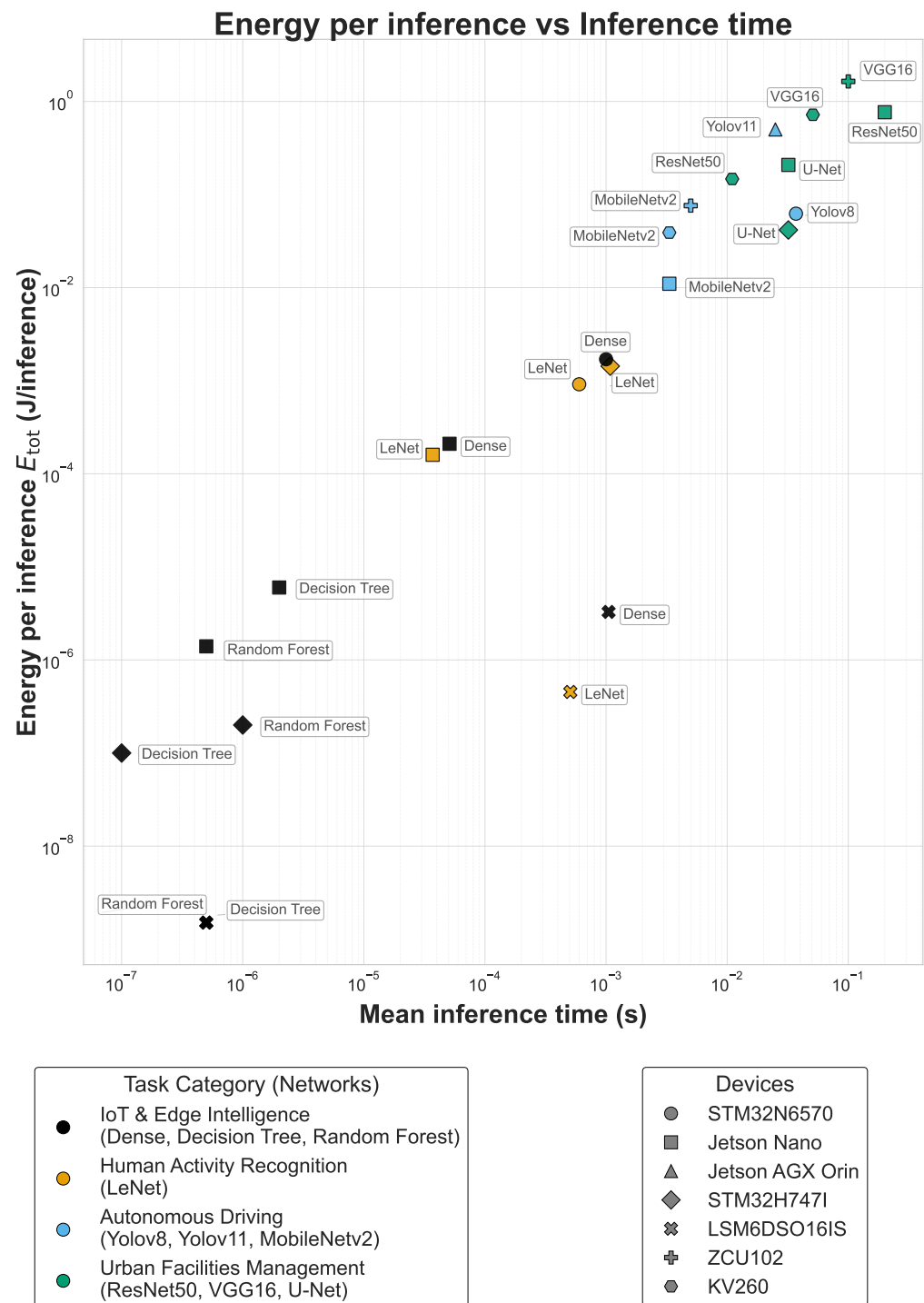


Figure 5. Energy per inference versus end-to-end inference time for all evaluated model–platform pairs. Within each hardware class, points tend to follow characteristic scaling trends; differences across classes highlight distinct design trade-offs between latency and energy.

Several key insights have emerged from this cross-platform comparison.

- Platform-specific scaling: Platforms with similar architectural characteristics (e.g., ISPU and STM32N6) follow comparable energy–time scaling trends, suggesting that underlying hardware architecture strongly influences efficiency characteristics.
- Manufacturing process impact: Older technologies (e.g., STM32H747I-DISCO with larger process nodes) exhibit significantly higher energy consumption relative to inference time than newer platforms fabricated with more advanced processes.

- Workload-specific clustering: Different types of neural network workloads (classification, detection, segmentation) form distinct clusters in the energy–time space, indicating that task requirements should inform hardware selection.
- Non-linear scaling: The energy cost of reducing inference time increases non-linearly across all platforms, with diminishing returns as platforms approach their performance limits.

These observations highlight the importance of aligning hardware capabilities with application needs in smart city AI deployments. For always-on sensing applications prioritizing ultra-low-power consumption over latency, ISPU-class devices excel in energy efficiency. STM32N6-class platforms offer a well-balanced solution for moderate power limitations and performance. NVIDIA Jetson and similar GPU-accelerated systems supply the computational power required for demanding real-time applications, though they do incur considerably higher energy usage.

Table 3. Mean power $\bar{P}$ during the inference window, end-to-end latency T , energy per inference E (integrated over the same window), and $EDP = E \cdot T$. Uncertainties report $\pm 2\sigma$ over $N = 30$ repeated runs.

Board	Network	Power (W)	$2\sigma_P$	Time (s)	$2\sigma_T$	Energy (J)	EDP (J·s)
LSM6DSO16IS	Random Forest	3.0×10^{-3}	6×10^{-4}	5.0×10^{-7}	1×10^{-7}	1.5×10^{-9}	7.5×10^{-16}
	LeNet	8.9×10^{-4}	2×10^{-4}	5.1×10^{-4}	3×10^{-5}	4.5×10^{-7}	2.3×10^{-10}
	Dense	3.1×10^{-3}	6×10^{-4}	1.1×10^{-3}	6×10^{-5}	1.4×10^{-6}	1.6×10^{-9}
	Decision Tree	3.0×10^{-3}	5×10^{-4}	5.0×10^{-7}	1×10^{-7}	1.5×10^{-9}	7.5×10^{-16}
STM32H747I DISCO	Random Forest	2.7×10^{-1}	3×10^{-4}	1.0×10^{-6}	5×10^{-8}	2.0×10^{-7}	2.0×10^{-13}
	Decision Tree	9.2×10^{-1}	1×10^{-4}	1.0×10^{-7}	5×10^{-8}	1.0×10^{-7}	1.0×10^{-14}
	LeNet	1.3×10^0	1×10^{-3}	1.1×10^{-3}	6×10^{-5}	1.4×10^{-3}	1.6×10^{-6}
	U-Net	1.3×10^0	1×10^{-3}	3.2×10^{-2}	2×10^{-3}	4.2×10^{-2}	1.3×10^{-3}
STM32N6570 DK	Dense	1.7×10^0	9×10^{-2}	1.0×10^{-3}	2×10^{-5}	1.7×10^{-3}	1.7×10^{-6}
	LeNet	1.6×10^0	8×10^{-2}	6.0×10^{-4}	1×10^{-5}	9.2×10^{-4}	5.5×10^{-7}
	Yolov8	1.7×10^0	9×10^{-2}	3.7×10^{-2}	7×10^{-4}	6.2×10^{-2}	2.3×10^{-3}
Jetson Nano	Decision Tree	3.0×10^0	3×10^{-1}	2.0×10^{-6}	1×10^{-6}	6.0×10^{-6}	1.2×10^{-11}
	Dense	4.1×10^0	4×10^{-1}	5.1×10^{-5}	1×10^{-6}	2.1×10^{-4}	1.1×10^{-8}
	LeNet	4.4×10^0	4×10^{-1}	3.7×10^{-5}	1×10^{-6}	1.6×10^{-4}	5.9×10^{-9}
	MobileNetv2	3.4×10^0	3×10^{-1}	3.3×10^{-3}	7×10^{-5}	1.1×10^{-2}	3.7×10^{-5}
	Random Forest	2.9×10^0	3×10^{-1}	5.0×10^{-7}	1×10^{-6}	1.4×10^{-6}	7.0×10^{-13}
	ResNet50	3.8×10^0	4×10^{-1}	2.0×10^{-1}	4×10^{-3}	7.7×10^{-1}	1.5×10^{-1}
	U-Net	6.5×10^0	7×10^{-1}	3.2×10^{-2}	6×10^{-4}	2.1×10^{-1}	6.7×10^{-3}
Jetson Orin	Yolov11	2.0×10^1	1×10^0	2.5×10^{-2}	5×10^{-4}	5.0×10^{-1}	1.2×10^{-2}
ZCU102	MobileNetv2	1.4×10^1	8×10^{-1}	5.0×10^{-3}	5×10^{-4}	7.6×10^{-2}	3.8×10^{-4}
	VGG16	1.2×10^1	7×10^{-1}	1.0×10^{-1}	1×10^{-3}	1.6×10^0	1.6×10^{-1}
KV260	MobileNetv2	1.2×10^1	6×10^{-1}	3.3×10^{-3}	3×10^{-4}	3.9×10^{-2}	1.3×10^{-4}
	ResNet50	1.3×10^1	7×10^{-1}	1.1×10^{-2}	1×10^{-4}	1.5×10^{-1}	1.6×10^{-3}
	VGG16	1.4×10^1	7×10^{-1}	5.1×10^{-2}	5×10^{-4}	7.2×10^{-1}	3.7×10^{-2}

5. Guidelines for IoT-Driven Smart City Applications

The experimental results discussed in Section 4 show that the considered platforms occupy very different regions of the energy–latency design space: from sub-*mW*-scale intelligent sensors to multi-watt embedded GPUs. In this section, these measurements are translated into practical guidelines for smart city designers, explaining how different

device classes align with different inference tasks and how this addresses the research gap highlighted in the related work (Table 1).

Following Figure 1, the discussion is structured around four macro-categories of tasks that are recurrent in urban AI deployments: Human Activity Recognition, IoT and Edge Intelligence, Urban Facility Management, and Autonomous Driving and Urban Mobility. Each workload in the benchmark suite (gesture recognition, dense and convolutional classification, segmentation, detection) can be naturally mapped to one of these domains.

For Human Activity Recognition, such as gesture-based interaction with public terminals, context detection in wearables, or operator monitoring, the picture is clear. Gesture-recognition models can run at extremely low energy on the ISPU itself, making it the most natural choice for always-on, battery-powered devices that must react to simple patterns. When the number of gestures increases, or additional sensing is needed, MCU-class platforms offer a comfortable trade-off: they handle more complex models with very low latency while still staying within a manageable inference range. In contrast, FPGA and GPU boards are generally overkill for standalone Human Activity Recognition (HAR) and only become interesting if HAR is just one component of a larger perception stack already running on those devices.

The category of IoT and Edge Intelligence covers a broad family of tasks: simple image or signal classification, anomaly detection on sensor data, and filtering of events before they are forwarded to the cloud or computed on the edge. Measurements confirm that MCU+NPU platforms are particularly well-suited for this class of problems: quantized INT8 models deliver low energy per inference and millisecond latency. Traditional MCUs, such as STM32H7, remain viable but incur a penalty in both power consumption and inference time, especially on older technology nodes. ISPU devices, in this context, are best used as intelligent front-ends that pre-filter raw signals (for example, activity/no-activity flags) before they reach the MCU tier.

Urban Facility Management in modern cities increasingly relies on visual analytics: aerial inspection of roads and buildings, monitoring of vegetation, recognition of urban assets, smart grids, and even waste classification. Such models stress the memory system and benefit from higher compute density: this is where embedded GPUs become a natural choice, provided that the nodes have access to stable power. FPGA-based SoCs, such as AMD Zynq UltraScale+ MPSoC ZCU102, are attractive when real-time guarantees are required, or when custom sensor interfaces must be tightly integrated with the accelerator. MCU+NPU platforms can host compressed or scaled-down variants of these models, enabling segmentation or classification in very constrained devices at the price of reduced accuracy or resolution.

Finally, Autonomous Driving and Urban Mobility combines the most demanding workloads in terms of latency. In order to have low response time, embedded GPUs are currently the most practical solution at the edge. MCUs and ISPUs, in this picture, play a complementary role: they are well-suited for local triggering and pre-filtering, which reduce the duty cycle and bandwidth requirements of the high-end perception nodes.

To make these trends more explicit, Table 4 provides a qualitative summary of how each platform behaves across the main workloads used in the experimental evaluation: gesture recognition (HAR), semantic segmentation (U-Net), and object detection (YOLO). For each platform–workload pair, relative levels of energy per inference and latency are reported, using three coarse classes: Low (L), Medium (M), and High (H). These labels are built from the measured power, time, and energy values presented in Section 4, but are summarized for readability.

In practice, hardware selection can be driven by three inputs: (i) the task class, (ii) the latency target, and (iii) the available power budget and duty cycle. Figure 6 summarizes the corresponding decision flow based on the measured operating regions.

Table 4. Qualitative cross-platform comparison across benchmarked workloads. For each platform and workload, relative energy per inference (E) and latency (T) are reported as L = low, M = medium, and H = high, relative across platform classes, not within one platform family.

Platform	HAR (E/T)	Classification (E/T)	Segmentation (E/T)	Detection (E/T)
ISPU (LSM6DSO16IS)	L/L	L/H	-	-
STM32N6 (NPU INT8)	L/L	L/L	M/L	M/L
STM32H7 (MCU)	L/L	L/L	M/H	M/M
ZCU102 (FPGA + DPU)	M/M	M/M	H/M	H/M
KV260 (FPGA + DPU)	M/M	M/M	H/M	H/M
Jetson Nano (eGPU)	M/L	H/L	H/ L-M	H/L
Jetson AGX Orin (eGPU)	M/L	M/L	H/L	H/L

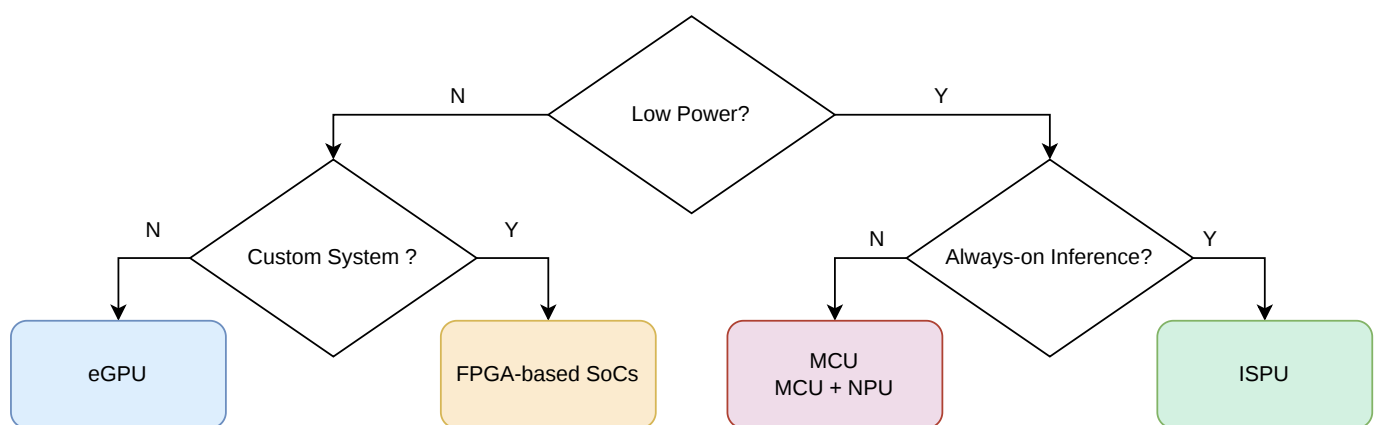


Figure 6. Measurement-driven deployment flowchart for IoT-driven smart-city applications. Decision nodes reflect the experimentally observed energy–latency regions and guide the selection of ISPU sensors, MCU (or MCU + NPU) platforms, embedded GPUs, and FPGA SoCs across typical workloads.

The above analysis can be summarized as follows:

- Embedded GPUs are the workhorses for heavy visual analytics: when there are multiple high-resolution streams, when latency constraints are tight, and when power is available, an NVIDIA Jetson-class device is the pragmatic choice. Model compression and INT8 quantization, as used in the experimental evaluation, are essential to keep the energy per inference under control.
- FPGA-based SoCs are most attractive in urban scenarios where deterministic latency, custom I/O, and tight integration with existing hardware pipelines are required (e.g., sensor interfacing, pre-processing in logic, or co-locating control and inference). Compared to embedded GPUs, their advantage is often system-level integration and timing guarantees rather than universally lower end-to-end energy for every model.
- MCU and MCU+NPU platforms are the natural backbone of the urban edge. The results indicate that NPU-accelerated devices, such as the STM32N6, consistently provide the best balance between latency and energy per inference for these workloads. At the same time, classic MCUs remain a solid option where NPUs are not yet available.
- ISPU-class sensors occupy the extreme low-power end of the spectrum. They are not substitutes for GPUs or MCUs, but they are uniquely suited to keep a persistent simple ML task with low footprint in urban scenarios.

This work provides a coherent measurement-based framework for reasoning about energy and performance. By grounding smart city tasks in concrete workloads and profiling them, a technological map is provided to help engineers decide, for each application, in which region of the hardware space they should operate and how far they can push energy efficiency without sacrificing functionality.

6. Conclusions and Discussion

Understanding and controlling the energy footprint of AI systems is particularly challenging in dynamic urban environments [14,66–68], yet it is a prerequisite for designing sustainable smart city infrastructures. This work presents a horizontal, measurement-driven approach that systematically profiles the energy–latency behavior of AI workloads across a spectrum of edge platforms. Across the evaluated configurations, measured inference power and energy span multiple orders of magnitude, confirming that smart-city AI deployments cannot be optimized with a one-size-fits-all platform choice and instead require tier-aware design.

The experimental characterization indicates that GPU-accelerated platforms, such as NVIDIA Jetson devices, are the most practical option for smart city analytics and AI services dominated by high-complexity vision pipelines and tight latency constraints. These systems benefit from mature software stacks (CUDA and TensorRT) and advanced hardware integration, enabling effective optimizations such as INT8 quantization and highly parallel kernels. However, their higher baseline power makes them best suited to grid-powered installations or high-duty-cycle workloads where the platform can be kept efficiently utilized.

At the opposite end, microcontroller-class devices equipped with dedicated accelerators (e.g., MCU+NPU) and ISPU-class sensors enable always-on intelligence under strict energy budgets. By performing local inference and early filtering, these tiers can reduce communication and data-movement costs and can trigger higher-end perception nodes only when needed, thereby lowering the overall duty cycle of energy-hungry platforms. In particular, NPU-enabled MCUs provide a robust “middle tier”: they preserve millisecond-scale responsiveness while keeping energy per inference compatible with battery-powered deployments.

Building on these measurements, the task clustering in Figure 1 and the guidelines in Section 5 translate raw power–latency numbers into design-oriented insights. Rather than identifying a single “best” platform, the results make explicit that different smart-city domains map naturally to different hardware tiers, and that optimal deployments often combine multiple tiers (ISPU/MCU/FPGA/GPU) along the sensor-to-edge continuum.

Several open challenges emerge from the scope constraints discussed in Section 3.7. First, further standardization of power-profiling methodologies and benchmark suites for heterogeneous AI systems would strengthen reproducibility and improve cross-study comparability. Second, the proposed benchmarking workflow is inherently extensible to emerging platforms (e.g., RISC-V-based accelerators and next-generation NPUs), as it relies on reproducible workload definitions and system-level energy–latency measurements that can be replicated across hardware tiers. Third, long-duration evaluations under realistic thermal and DVFS dynamics remain necessary to quantify drift over multi-hour deployments, especially on OS-based platforms.

Future research should also extend measurement boundaries beyond on-device inference to account for sensing and communication energy in end-to-end edge–cloud pipelines, since networking can dominate energy in some smart city scenarios and may shift the optimal design point toward more local processing. On the algorithmic side, investigating the power impact of emerging model families and exploiting dynamic adaptation mech-

anisms (e.g., early-exit inference, conditional computation, and resolution scaling) can further improve efficiency under real-world duty cycles. Finally, energy-aware neural architecture search and training objectives that treat energy, latency, and accuracy as co-optimized targets remain promising directions toward smarter and greener AI deployments in urban ecosystems.

Author Contributions: Conceptualization, S.B., F.F. and A.C.; methodology, S.B., F.F. and A.C.; software, S.B. and F.F.; validation, S.B. and F.F.; writing—original draft preparation, S.B. and F.F.; writing—review and editing, A.C.; supervision, A.C.; project administration, A.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been partially supported by Spoke 1 “FutureHPC & BigData” of ICSC—Centro Nazionale di Ricerca in High-Performance-Computing, Big Data and Quantum Computing, funded by European Union—NextGenerationEU, as well as by PRIN project “RETICULATE”, funded by the Italian Ministry of University (MUR).

Data Availability Statement: The original data presented in the study are openly available in: <https://www.kaggle.com/datasets/salvatorebramante/ispu-dat>, accessed on 6 March 2026.

Acknowledgments: Claude Opus 4.6 and ChatGPT 5.2 Thinking were used to revise the grammar, the style, and the overall flow of the text. Note that no technical content was generated with AI tools.

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

References

- Merenda, M.; Porcaro, C.; Iero, D. Edge Machine Learning for AI-Enabled IoT Devices: A Review. *Sensors* **2020**, *20*, 2533. [\[CrossRef\]](#)
- Wolniak, R.; Stecula, K. Artificial Intelligence in Smart Cities—Applications, Barriers, and Future Directions: A Review. *Smart Cities* **2024**, *7*, 1346–1389. [\[CrossRef\]](#)
- Farzaneh, H.; Malehmirchegini, L.; Bejan, A.; Afolabi, T.; Mulumba, A.; Daka, P.P. Artificial Intelligence Evolution in Smart Buildings for Energy Efficiency. *Appl. Sci.* **2021**, *11*, 763. [\[CrossRef\]](#)
- Rojek, I.; Mikołajewski, D.; Galas, K.; Piszcz, A. Advanced Deep Learning Algorithms for Energy Optimization of Smart Cities. *Energies* **2025**, *18*, 407. [\[CrossRef\]](#)
- Shuvo, M.M.H.; Islam, S.K.; Cheng, J.; Morshed, B.I. Efficient Acceleration of Deep Learning Inference on Resource-Constrained Edge Devices: A Review. *Proc. IEEE* **2023**, *111*, 42–91. [\[CrossRef\]](#)
- Cecilia, J.M.; Cano, J.C.; Morales-García, J.; Llanes, A.; Imbernón, B. Evaluation of Clustering Algorithms on GPU-Based Edge Computing Platforms. *Sensors* **2020**, *20*, 6335. [\[CrossRef\]](#) [\[PubMed\]](#)
- Kalapothis, S.; Flamis, G.; Kitsos, P. Efficient Edge-AI Application Deployment for FPGAs. *Information* **2022**, *13*, 279. [\[CrossRef\]](#)
- Mir, U.; Abbasi, U.; Mir, T.; Kanwal, S.; Alamri, S. Energy Management in Smart Buildings and Homes: Current Approaches, a Hypothetical Solution, and Open Issues and Challenges. *IEEE Access* **2021**, *9*, 94132–94148. [\[CrossRef\]](#)
- Boukerche, A.; Tao, Y.; Sun, P. Artificial intelligence-based vehicular traffic flow prediction methods for supporting intelligent transportation systems. *Comput. Netw.* **2020**, *182*, 107484. [\[CrossRef\]](#)
- Fanariotis, A.; Orphanoudakis, T.; Kotrotsios, K.; Fotopoulos, V.; Keramidas, G.; Karkazis, P. Power Efficient Machine Learning Models Deployment on Edge IoT Devices. *Sensors* **2023**, *23*, 1595. [\[CrossRef\]](#)
- Esfandi, S.; Tayebi, S.; Byrne, J.; Taminiau, J.; Giyahchi, G.; Alavi, S.A. Smart Cities and Urban Energy Planning: An Advanced Review of Promises and Challenges. *Smart Cities* **2024**, *7*, 414–444. [\[CrossRef\]](#)
- Rodriguez, C.; Degioanni, L.; Kameni, L.; Vidal, R.; Neglia, G. Evaluating the Energy Consumption of Machine Learning: Systematic Literature Review and Experiments. *arXiv* **2024**, arXiv:2408.15128. [\[CrossRef\]](#)
- Desislavov, R.; Martínez-Plumed, F.; Hernández-Orallo, J. Trends in AI inference energy consumption: Beyond the performance-vs-parameter laws of deep learning. *Sustain. Comput. Inform. Syst.* **2023**, *38*, 100857. [\[CrossRef\]](#)
- Alizadeh, N.; Castor, F. Green AI: A Preliminary Empirical Study on Energy Consumption in DL Models Across Different Runtime Infrastructures. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*; ACM: New York, NY, USA, 2024; pp. 134–139.
- Ning, Z.; Vandersteegen, M.; Van Beeck, K.; Goedemé, T.; Vandewalle, P. Power Consumption Benchmark for Embedded AI Inference. In *Proceedings of the International Conferences on Applied Computing 2024 and www/Internet 2024*, Zagreb, Croatia, 26–28 October 2024.

16. Lema, D.G.; Usamentiaga, R.; García, D.F. Quantitative comparison and performance evaluation of deep learning-based object detection models on edge computing devices. *Integration* **2024**, *95*, 102127. [CrossRef]
17. Isenkul, M.E. Energy-aware deep learning for real-time video analysis through pruning, quantization, and hardware optimization. *J.-Real-Time Image Process.* **2025**, *22*, 125. [CrossRef]
18. Amin, R.A.; Hasan, M.; Wiese, V.; Obermaisser, R. FPGA-Based Real-Time Object Detection and Classification System Using YOLO for Edge Computing. *IEEE Access* **2024**, *12*, 73268–73278. [CrossRef]
19. Tatar, G.; Bayar, S. Energy efficiency assessment in advanced driver assistance systems with real-time image processing on custom Xilinx DPUs. *J.-Real-Time Image Process.* **2024**, *21*, 157. [CrossRef]
20. Ghielmetti, N.; Loncar, V.; Pierini, M.; Roed, M.; Summers, S.; Aarrestad, T.; Petersson, C.; Linander, H.; Ngadiuba, J.; Lin, K.; et al. Real-time semantic segmentation on FPGAs for autonomous vehicles with hls4mL. *Mach. Learn. Sci. Technol.* **2022**, *3*, 045011. [CrossRef]
21. Sciangula, G.; Restuccia, F.; Biondi, A.; Buttazzo, G. Hardware Acceleration of Deep Neural Networks for Autonomous Driving on FPGA-based SoC. In Proceedings of the 2022 25th Euromicro Conference on Digital System Design (DSD), Maspalomas, Spain, 31 August–2 September 2022; pp. 406–414.
22. Baczmanski, M.; Wasala, M.; Kryjak, T. Implementation of a Perception System for Autonomous Vehicles Using a Detection-Segmentation Network in SoC FPGA. In *Proceedings of the Applied Reconfigurable Computing. Architectures, Tools, and Applications*; Palumbo, F., Keramidas, G., Voros, N., Diniz, P.C., Eds.; Springer Nature: Cham, Switzerland, 2023; pp. 200–211.
23. Latotzke, C.; Ciesielski, T.; Gemmeke, T. Design of High-Throughput Mixed-Precision CNN Accelerators on FPGA. *arXiv* **2022**, arXiv:2208.04854.
24. Borrás, H.; Guglielmo, G.D.; Duarte, J.; Ghielmetti, N.; Hawks, B.; Hauck, S.; Hsu, S.C.; Kastner, R.; Liang, J.; Meza, A.; et al. Open-source FPGA-ML codesign for the MLPerf Tiny Benchmark. *arXiv* **2022**, arXiv:2206.11791.
25. Iqbal, O.; Muro, V.I.T.; Katoch, S.; Spanias, A.; Jayasuriya, S. Adaptive Subsampling for ROI-Based Visual Tracking: Algorithms and FPGA Implementation. *IEEE Access* **2022**, *10*, 90507–90522. [CrossRef]
26. Sun, L.; Muguira, L.; Jiménez, J.; Lázaro, J.; Yong, W. High Performance Platform to Detect Faults in the Smart Grid by Artificial Intelligence Inference. *IEEE Trans. Smart Grid* **2024**, *15*, 504–512. [CrossRef]
27. Vaithianathan, M.; Patil, M.; Ng, S.F.; Udkar, S. Comparative Study of FPGA and GPU for High-Performance Computing and AI. *Esp Int. J. Adv. Comput. Technol. (Esp-Ijact)* **2024**, *1*, 37–46.
28. Kumar, A.; Goyal, S.; Varma, M. Resource-efficient Machine Learning in 2 KB RAM for the Internet of Things. In *Proceedings of the 34th International Conference on Machine Learning*; 2017; pp. 1935–1944. Available online: <https://proceedings.mlr.press/> (accessed on 5 March 2026).
29. Gupta, C.; Suggala, A.S.; Goyal, A.; Simhadri, H.V.; Paranjape, B.; Kumar, A.; Goyal, S.; Udupa, R.; Varma, M.; Jain, P. ProtoNN: Compressed and Accurate kNN for Resource-scarce Devices. In *Proceedings of the 34th International Conference on Machine Learning*; 2017; pp. 1331–1340. Available online: <https://proceedings.mlr.press/v70/gupta17a.html> (accessed on 5 March 2026).
30. Pau, D.P.; Tognocchi, S.; Marcon, M.; Pau, D.P.; Tognocchi, S.; Marcon, M. Learning Online MEMS Calibration with Time-Varying and Memory-Efficient Gaussian Neural Topologies. *Sensors* **2025**, *25*, 3679. [CrossRef]
31. Chowdhary, M.; Saha, S.S. On-Sensor Online Learning and Classification Under 8 KB Memory. In Proceedings of the 2023 26th International Conference on Information Fusion (FUSION), Charleston, SC, USA, 27–30 June 2023; pp. 1–8.
32. Messina, A.; Lazzaro, A.; Carotenuto, R.; Merenda, M. Preliminary analysis of the exploitation of QVAR sensor for gesture recognition. In Proceedings of the 2025 10th International Conference on Smart and Sustainable Technologies (SpliTech), Bol and Split, Croatia, 16–20 June 2025; pp. 1–7.
33. Nguyen, J.; Grimsley, R.; Iannucci, B. TrafficNNode: Low Power Vehicle Sensing Platform for Smart Cities. In Proceedings of the 2021 IEEE International Conference on Smart Internet of Things (SmartIoT), Virtual Conference, 13–15 August 2021; pp. 278–282.
34. Son, S.; Seo, A.; Eo, G.; Gill, K.; Gong, T.; Kim, H.S.; Son, S.; Seo, A.; Eo, G.; Gill, K.; et al. MiCrowd: Vision-Based Deep Crowd Counting on MCU. *Sensors* **2023**, *23*, 3586. [CrossRef]
35. Fanariotis, A.; Orphanoudakis, T.; Fotopoulos, V.; Fanariotis, A.; Orphanoudakis, T.; Fotopoulos, V. Reducing the Power Consumption of Edge Devices Supporting Ambient Intelligence Applications. *Information* **2024**, *15*, 161. [CrossRef]
36. Shah, J.; Mishra, B. IoT enabled environmental monitoring system for smart cities. In Proceedings of the 2016 International Conference on Internet of Things and Applications (IOTA), Pune, India, 22–24 January 2016; pp. 383–388.
37. Sechi, F.; Fanucci, L.; Luschi, S.; Perini, S.; Madesani, M. Design of a Distributed Embedded System for Domotic Applications. In *Proceedings of the 2008 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools*; IEEE Computer Society: Los Alamitos, CA, USA, 2008; pp. 427–431.
38. Dubey, S.; Singh, P.; Yadav, P.; Singh, K.K. Household Waste Management System Using IoT and Machine Learning. *Procedia Comput. Sci.* **2020**, *167*, 1950–1959. [CrossRef]
39. Sallang, N.C.A.; Islam, M.T.; Islam, M.S.; Arshad, H. A CNN-Based Smart Waste Management System Using TensorFlow Lite and LoRa-GPS Shield in Internet of Things Environment. *IEEE Access* **2021**, *9*, 153560–153574. [CrossRef]

40. Olawade, D.B.; Fapohunda, O.; Wada, O.Z.; Usman, S.O.; Ige, A.O.; Ajisafe, O.; Oladapo, B.I. Smart waste management: A paradigm shift enabled by artificial intelligence. *Waste Manag. Bull.* **2024**, *2*, 244–263. [\[CrossRef\]](#)
41. Vitek, S.; Melničuk, P. A Distributed Wireless Camera System for the Management of Parking Spaces. *Sensors* **2017**, *18*, 69. [\[CrossRef\]](#)
42. Chen, L.C.; Sheu, R.K.; Peng, W.Y.; Wu, J.H.; Tseng, C.H.; Chen, L.C.; Sheu, R.K.; Peng, W.Y.; Wu, J.H.; Tseng, C.H. Video-Based Parking Occupancy Detection for Smart Control System. *Appl. Sci.* **2020**, *10*, 1079. [\[CrossRef\]](#)
43. Ke, R.; Zhuang, Y.; Pu, Z.; Wang, Y. A Smart, Efficient, and Reliable Parking Surveillance System With Edge Artificial Intelligence on IoT Devices. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 4962–4974. [\[CrossRef\]](#)
44. Amato, G.; Carrara, F.; Falchi, F.; Gennaro, C.; Meghini, C.; Vairo, C. Deep learning for decentralized parking lot occupancy detection. *Expert Syst. Appl.* **2017**, *72*, 327–334. [\[CrossRef\]](#)
45. Barthélemy, J.; Verstaavel, N.; Forehead, H.; Perez, P.; Barthélemy, J.; Verstaavel, N.; Forehead, H.; Perez, P. Edge-Computing Video Analytics for Real-Time Traffic Monitoring in a Smart City. *Sensors* **2019**, *19*, 2048. [\[CrossRef\]](#) [\[PubMed\]](#)
46. Belcastro, L.; Marozzo, F.; Orsino, A.; Talia, D.; Trunfio, P. Edge-Cloud Continuum Solutions for Urban Mobility Prediction and Planning. *IEEE Access* **2023**, *11*, 38864–38874. [\[CrossRef\]](#)
47. Yang, H.F.; Cai, J.; Liu, C.; Ke, R.; Wang, Y. Cooperative multi-camera vehicle tracking and traffic surveillance with edge artificial intelligence and representation learning. *Transp. Res. Part C Emerg. Technol.* **2023**, *148*, 103982. [\[CrossRef\]](#)
48. Noh, J.; Jo, Y.; Kim, J.; Min, K. Enhancing Transportation Safety with Infrastructure Cooperative Autonomous Driving System. *Int. J. Automot. Technol.* **2024**, *25*, 61–69. [\[CrossRef\]](#)
49. Torres, P.; Marques, H.; Marques, P.; Torres, P.; Marques, H.; Marques, P. Pedestrian Detection with LiDAR Technology in Smart-City Deployments—Challenges and Recommendations. *Computers* **2023**, *12*, 65. [\[CrossRef\]](#)
50. Sarvajcz, K.; Ari, L.; Menyhart, J.; Sarvajcz, K.; Ari, L.; Menyhart, J. AI on the Road: NVIDIA Jetson Nano-Powered Computer Vision-Based System for Real-Time Pedestrian and Priority Sign Detection. *Appl. Sci.* **2024**, *14*, 1440. [\[CrossRef\]](#)
51. Yazici, M.T.; Basurra, S.; Gaber, M.M.; Yazici, M.T.; Basurra, S.; Gaber, M.M. Edge Machine Learning: Enabling Smart Internet of Things Applications. *Big Data Cogn. Comput.* **2018**, *2*, 26. [\[CrossRef\]](#)
52. Badidi, E. Edge AI and Blockchain for Smart Sustainable Cities: Promise and Potential. *Sustainability* **2022**, *14*, 7609. [\[CrossRef\]](#)
53. Loseto, G.; Scioscia, F.; Ruta, M.; Gramegna, F.; Ieva, S.; Fasciano, C.; Bilenchi, I.; Loconte, D.; Sciascio, E.D. A Cloud-Edge Artificial Intelligence Framework for Sensor Networks. In *Proceedings of the 2023 9th International Workshop on Advances in Sensors and Interfaces (IWASI)*; IEEE: New York, NY, USA, 2023; pp. 149–154.
54. Chen, X.; Wen, H.; Ni, W.; Zhang, S.; Wang, X.; Xu, S.; Pei, Q. Distributed Online Optimization of Edge Computing With Mixed Power Supply of Renewable Energy and Smart Grid. *IEEE Trans. Commun.* **2022**, *70*, 389–403. [\[CrossRef\]](#)
55. Arif, A.; Barrigon, F.A.; Gregoretti, F.; Iqbal, J.; Lavagno, L.; Lazarescu, M.T.; Ma, L.; Palomino, M.; Segura, J.L.L. Performance and energy-efficient implementation of a smart city application on FPGAs. *J. Real-Time Image Process.* **2020**, *17*, 729–743. [\[CrossRef\]](#)
56. Balasubramanian, V.; Otoum, S.; Reisslein, M. VeNet: Hybrid Stacked Autoencoder Learning for Cooperative Edge Intelligence in IoV. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 16643–16653. [\[CrossRef\]](#)
57. Chakraborty, A.; Tavernier, W.; Kourtis, A.; Pickavet, M.; Oikonomakis, A.; Colle, D. Profiling Concurrent Vision Inference Workloads on NVIDIA Jetson. In *Proceedings of the 2025 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Ghent, Belgium, 11–13 May 2025; pp. 359–361.
58. Nigam, I.; Huang, C.; Ramanan, D. Ensemble Knowledge Transfer for Semantic Segmentation. In *Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Lake Tahoe, NV, USA, 12–15 March 2018; pp. 1499–1508.
59. Lin, T.; Maire, M.; Belongie, S.J.; Bourdev, L.D.; Girshick, R.B.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*; Springer International Publishing: Cham, Switzerland, 2014.
60. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
61. Bramante, S.; Ferrandino, F. ISPU Gesture Dataset. 2024. Available online: <https://www.kaggle.com/datasets/salvatorebramante/ispu-dat/> (accessed on 1 February 2026).
62. STmicroelectronics. Software Solution for MEMS Sensors with Graphical No-Code Design of Algorithms and Development of Embedded AI Features. Available online: <https://www.st.com/en/development-tools/mems-studio.html> (accessed on 1 February 2026).
63. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [\[CrossRef\]](#)
64. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:1409.1556. [\[CrossRef\]](#)
65. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv* **2015**, arXiv:1505.04597. [\[CrossRef\]](#)

66. Kasioulis, M.; Symeonides, M.; Ioannou, G.; Pallis, G.; Dikaiakos, M.D. Energy modeling of inference workloads with AI accelerators at the Edge: A benchmarking study. In Proceedings of the 2024 IEEE International Conference on Cloud Engineering (IC2E), Paphos, Cyprus, 24–27 September 2024; pp. 189–196.
67. Yang, T.J.; Chen, Y.H.; Emer, J.; Sze, V. A method to estimate the energy consumption of deep neural networks. In Proceedings of the 2017 51st Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, 29 October–1 November 2017; pp. 1916–1920.
68. Wang, L.; Zhang, X.; Su, H.; Zhu, J. A Comprehensive Survey of Continual Learning: Theory, Method and Application. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 5362–5383. [[CrossRef](#)] [[PubMed](#)]

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.