JOURNAL OF BAKU ENGINEERING UNIVERSITY- MATHEMATICS AND COMPUTER SCIENCE

2025. Volume 9, Number 2 *Pages 186-195*

UDC: 004.932.2

DOI: https://doi.org/10.30546/09090.2025.1010.2015

BENCHMARKING 2D AND 3D OBJECT DETECTION MODELS ON DIFFERENT EDGE COMPUTING PLATFORMS

Shahla URALOVA[1][0009-0004-0146-5173], Rasim MAHMUDOV[2][0009-0006-1385-1412], Amil BABAYEV[1][0009-0005-4823-6201] and Qurban YUSUFOV[1][0009-0004-0530-9709]

[1] *Cyber Security and Computer Engineering, Baku Engineering University, Baku, Azerbaijan*

suralova@std.beu.edu.az, amilb@beu.edu.az, quyusufov@beu.edu.az

[2] *Information Technology and Programming, Baku Engineering University, Baku, Azerbaijan rkmahmudov@gmail.com*

ARTICLE INFO

Article history:
Received: 2025-09-01
Received in revised form:2025-09-11
Accepted:2026-01-12
Available online

Keywords:
Edge computing, Convolutional neural networks, Benchmarking.

2010 Mathematics Subject Classifications: 68T07, 68T45

ABSTRACT

The deployment of artificial intelligence models on edge computing platforms is imperative for applications necessitating real-time perception, including object detection and gesture recognition. This paper presents a thoroughgoing benchmarking study of 2D and 3D convolutional neural network models across a range of representative edge devices, including the NVIDIA Jetson AGX Orin, Jetson Nano, and Raspberry Pi 3. The study utilizes gesture recognition and mask detection as the primary use cases for evaluating the performance of these models. We evaluated the inference time, memory footprint, and power consumption under varying input resolutions and sequence lengths, thereby capturing the latency–accuracy–energy trade-offs that govern practical deployment. The results show that AGX Orin consistently handles high-demand scenarios with low latency and stable throughput, whereas Jetson Nano offers competitive efficiency in energy-constrained settings. Raspberry Pi 3, while limited, establishes a useful baseline for CPU-only operation and highlights optimization needs. We also examined the effects of quantization and runtime optimizations and reported configurations that deliver robust performance without substantial accuracy loss. This study synthesizes these findings into actionable guidelines for selecting models, precisions, and device classes according to operational constraints. By providing a reproducible evaluation framework and device-aware recommendations, this study supports developers in balancing speed, accuracy, and energy efficiency, thereby enabling reliable real-time AI on resource-limited edge platforms. Our methodology uses warm-up iterations, repeated trials per configuration, synchronized power sampling, and pre-processing to ensure fair comparisons.

1 Introduction

In the evolving landscape of edge computing, the deployment of artificial intelligence (AI) models for real-time applications, such as object detection and gesture recognition, has become increasingly crucial. Unlike cloud-based approaches, where reliance on centralized servers often introduces delays, edge platforms offer the opportunity to process data locally, thereby enabling faster and more reliable inference. This study presents a comprehensive benchmarking of both 2D and 3D convolutional neural network (CNN) models across multiple edge devices, including NVIDIA's Jetson AGX Orin, Jetson Nano, and Raspberry Pi, to evaluate their performance under varying levels of computational complexity and input demand.

Traditional cloud-based solutions for AI inference often struggle with latency, bandwidth limitations, and privacy concerns, particularly in time-sensitive applications such as health monitoring, autonomous driving, and surveillance. Edge computing directly addresses these challenges by enabling on-device processing, reducing the dependency on external servers, and supporting real-time decision-making. The widespread deployment of IoT devices, smart cameras, and mobile robots further emphasizes the need for efficient and robust AI inference at the edge, making the selection of appropriate models and hardware platforms a critical research topic.

In this context, the present study investigates the performance of 3D CNN models for gesture recognition using the 20BN and Jester datasets, as well as 2D CNN and ResNet50 models for real-time mask detection using the MaskedFace-Net and RMFD datasets. The experiments are conducted on three representative edge devices: Jetson AGX Orin, Jetson Nano, and Raspberry Pi 3. The key specifications of these devices are summarized in Table 1. The differences in CPU/GPU architecture, memory capacity, and power requirements serve as a foundation for understanding how hardware constraints affect model performance.

Table 1. Specification of used edge devices.

Specification	AGX Orin	Jetson Nano	Raspberry Pi 3
CPU	8-core ARM Cortex-A78AE @ 2.188 GHz	4-core ARM Cortex-A57 @ 1.43 GHz	Quad-core ARM Cortex-A53 @ 1.2 GHz
GPU	2048 CUDA cores	128-core Maxwell GPU	Broadcom VideoCore IV
Memory	32 GB LPDDR5	4 GB LPDDR4	1 GB LPDDR2
Storage	64 GB eMMC	microSD	MicroSD
Power	15-60 W	5-10 W	5 W
JetPack / OS	JetPack 6.0	JetPack 4.2.1	N/A (Uses Raspbian)
Framework	TensorFlow, PyTorch	TensorFlow, PyTorch	TensorFlow, PyTorch
AI Performance	Up to 275 TOPS	0.5 TOPS	Limited AI capabilities

Although several benchmarking studies have explored deep neural network (DNN) deployment on embedded systems, most have focused either on single-model scenarios (e.g., only 2D CNNs) or single-device evaluations (e.g., Jetson Nano or TX2). Moreover, much of the prior work primarily emphasizes accuracy and inference speed while neglecting equally important considerations, such as memory footprint and energy efficiency. This gap underscores the need for a more comprehensive evaluation framework that captures the trade-offs between computational power, resource utilization, and energy consumption, thereby enabling informed decisions for practical edge AI deployments.

Key performance metrics, such as inference time, memory usage, and CPU and GPU power consumption, were systematically analyzed across these models and devices. This research highlights the trade-offs between computational power and energy efficiency, which are crucial for optimizing the deployment of AI models in resource-constrained environments. This benchmarking provides valuable insights into selecting appropriate hardware and model configurations for specific edge computing applications, focusing on achieving a balance between speed, accuracy, and power consumption of the model.

Here are the main contributions of this paper:

- We present a comprehensive benchmarking of 3D CNN models for gesture recognition using the 20BN and Jester datasets, evaluating their scalability across different frame input sizes.
- We evaluate 2D CNN and ResNet50 models for face mask detection using the Masked Face-Net and RMFD datasets, providing insights into their robustness under varying image resolutions.
- We analyze and compare edge devices, including Jetson AGX Orin, Jetson Nano, and Raspberry Pi 3, focusing on inference time, memory usage, and power consumption.
- We provide practical guidelines for deploying AI models on edge platforms, highlighting trade-offs between speed, accuracy, and energy efficiency that are critical for real-world applications.

This comprehensive set of contributions provides a robust benchmarking for understanding AI model deployments on edge computing platforms.

2 Related work

In recent years, many studies have benchmarked the deployment of deep neural networks (DNNs) on embedded/edge hardware, especially the NVIDIA Jetson family (Nano, TX2, Xavier/AGX, Orin), for image classification, object detection, face/gesture recognition, and scene understanding. Prior studies typically report end-to-end performance under strict compute and power budgets and contrast 2D vs. 3D CNN architectures, runtime stacks (TensorRT, ONNX Runtime, PyTorch/TensorFlow), and model-level optimizations such as quantization/pruning and kernel fusion [1–5], [7–9]. However, evaluation protocols vary widely across studies terms of metrics (latency/throughput, accuracy, memory footprint, CPU/GPU power draw) and experimental knobs (input resolution, sequence length, batch size, device power/thermal modes), which complicates cross-paper comparison despite valuable insights [1–5], [15]. Therefore, we motivate a unified benchmark contrasting 3D CNNs for gesture recognition and 2D CNN/ResNet-50 for mask detection on common edge devices, with consistent measurements of inference time, memory usage, and CPU/GPU power. Techniques such as approximations [6], lightweight architectures like MobileNetV2 [8] and MobileNetV3 [9], and quantization/pruning have been proposed to reduce compute and memory overhead for embedded AI.

2.1 DNN Deployment on Embedded Boards

A substantial line of work evaluates DNN workloads on resource-constrained embedded boards, with particular attention to Jetson devices. Süzen et al. compare Jetson Nano, Jetson TX2, and Raspberry Pi 4 using a CNN for fashion-product classification, reporting power consumption, GPU/CPU/RAM utilization, accuracy, and total cost across datasets from 5k to 45k images [1]. Complementing device-level comparisons, EdgeFace targets face recognition under edge constraints and shows that careful architectural choices can retain accuracy while improving computational efficiency for small form-factor boards [2]. Broader edge-class benchmarking on phone-grade SoCs likewise links memory bandwidth and on-device acceleration to end-to-end throughput [15], while direct Nano vs. Raspberry Pi studies emphasize the role of CUDA-capable GPUs for real-time inference at moderate resolutions. Concurrently, model-side techniques such as parameter sharing, approximations, and quantification/pruning are employed to minimize computing and memory expenditures. [6], [8], [9].

Relation to our study. These works establish feasibility and highlight cost/power trade-offs, but typically examine single tasks or device classes. We extend this line by using a unified protocol over heterogeneous devices (AGX Orin, Nano, RPi 3) and two application families (gesture, mask detection), while jointly reporting latency, memory, and CPU/GPU power.

2.2 Learning Spatiotemporal Features with 3D CNNs for Gesture Recognition

Beyond 2D image pipelines, several studies explore 3D inference and spatiotemporal or geometric modeling on embedded hardware. Ullah and Kim [3] benchmark the Jetson platform for 3D point-cloud and hyperspectral classification, emphasizing that memory bandwidth and on-chip acceleration critically impact throughput when input dimensionality grows. Similarly, Choe et al. [4] analyze 3D object detectors on Jetson devices, showing that detector capacity, input resolution, and kernel-level optimizations (e.g., inference runtimes and deployment stacks) jointly determine real-time viability. Collectively, these results reinforce a common observation: 3D workloads impose heavier compute and memory pressure than comparable 2D pipelines, tightening power and thermal margins on embedded GPUs. Early works such as C3D [10] and I3D [11] demonstrated the effectiveness of 3D CNNs for spatiotemporal feature learning, though their computational cost remains a challenge on edge devices.

Relation to our study. Our benchmark brings this perspective to video-based gesture recognition with 3D CNNs, contrasting it against 2D CNN pipelines under matched measurement settings to quantify the cost of temporal modeling.

2.3 AI for Health Safety Measures in Edge Computing

Edge-resident perception has also been explored for public-health and safety applications. Jovović et al. demonstrate face-mask detection on Jetson-class devices, integrating ML with IoT to enable responsive, on-device screening in pandemic scenarios [5]. Widely used mask datasets include MaskedFace-Net, which distinguishes correctly and incorrectly worn masks [13], and the real-world RMFD collection [14], both enabling reproducible assessment of lightweight 2D CNNs and higher-capacity backbones such as ResNet-50 [7].

Relation to our study. We evaluate ResNet-50 on MaskedFace-Net and a lighter 2D CNN on RMFD, directly measuring the latency–memory–power envelope that governs deployability of mask-detection models on edge hardware.

2.4 Summary and Research Gap

Across [1–5],[10–15], prior work often focuses on single-model families or single devices and primarily reports accuracy/latency, while memory footprint and CPU/GPU power draw are covered less systematically. In addition, protocol heterogeneity—datasets, input sizes, runtimes—complicates cross-paper comparison. Unlike ad-hoc reports, we hold preprocessing, runtime stack, and power settings constant across devices [15], enabling fair attribution of performance differences to model capacity and input complexity. Our study addresses these limitations with a task-balanced, cross-device benchmark that (i) spans 3D gesture and 2D mask detection pipelines, (ii) enforces consistent measurement (batch size, warm-up, metrics), and (iii) reports inference time, peak memory, and CPU/GPU power draw together to surface actionable deployment trade-offs.

3 Benchmark analysis

3.1 Experimental Setup

We conduct a comprehensive evaluation of two representative application families—specifically gesture recognition and face-mask detection—across three distinct embedded edge computing devices: the NVIDIA Jetson AGX Orin, Jetson Nano, and Raspberry Pi 3. Our benchmark methodology encompasses both 3D and 2D inference regimes and implements a carefully standardized protocol to guarantee fair, reproducible, and meaningful comparisons across these diverse hardware platforms. Figure 1 provides a detailed overview of the experimental setup, including the tasks being evaluated, the model architectures employed (specifically 3D CNNs for temporal analysis and both lightweight and ResNet-based 2D CNNs for spatial analysis), the datasets utilized for training and validation (including 20BN-Jester, Jester, MaskedFace-Net, and RMFD), and the comprehensive range of input dimensions strategically selected to simulate both computationally light and heavy workloads. In the subsequent sections of this paper, we elaborate on the precise resolutions, sequence lengths, runtime environments, and measurement procedures implemented throughout our testing process. The following text is intended to provide a comprehensive overview of the subject matter. The methodological elements outlined above collectively establish a thorough and systematic blueprint for accurately evaluating critical performance metrics. These metrics include processing latency, memory utilization footprint, and power consumption characteristics. The blueprint effectively isolates the intrinsic hardware capabilities of each device from potentially confounding software implementation differences. For gesture recognition, we used the 20BN and Jester datasets [12], which are widely adopted for benchmarking hand-gesture classification.

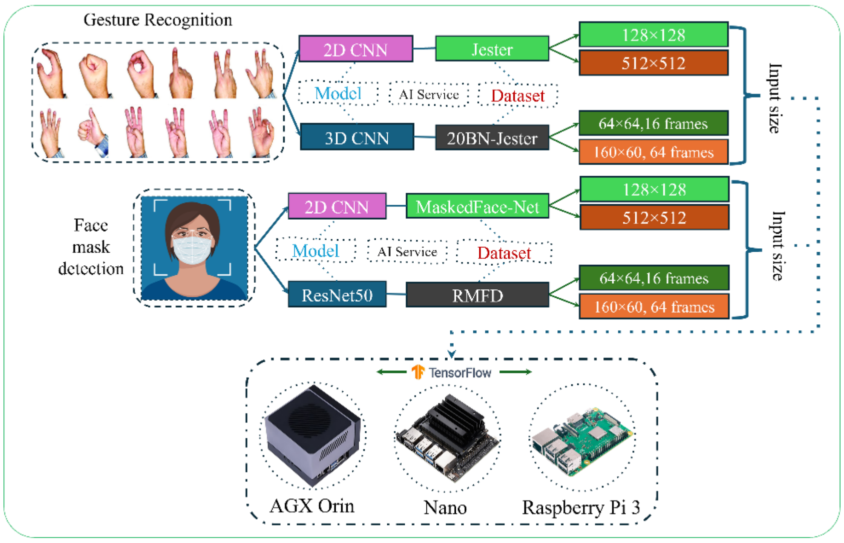

Fig. 1. Architecture of benchmarking system

For mask detection, we evaluate two model families: a lightweight 2D CNN on the RMFD dataset (inputs at 300 × 300 and 516 × 516), and a ResNet-50 on the MaskedFace-Net dataset (224 × 224 and 960 × 544). This combination reflects both constrained and computationally demanding pipelines, and allows us to capture how model capacity interacts with device limitations.

All devices were configured in performance-oriented modes to reduce throttling and stabilize results. On Jetson boards, experiments were run under JetPack environments (v6.0 for AGX Orin and v4.2.1 for Nano), while the Raspberry Pi used Raspbian with TensorFlow/PyTorch backends.

We locked GPU clocks where possible to minimize DVFS-related variance. Unless otherwise specified, batch size = 1 and single-stream inference were used.

Measurements were taken under consistent conditions. Latency is reported as the median inference time over repeated runs after discarding warm-up iterations. Memory usage reflects peak resident set size (RSS) during inference. CPU and GPU power draw were recorded through on-device telemetry (tegrastats for Jetson boards and vcgencmd for Raspberry Pi) at 1 Hz sampling, then averaged. To account for runtime variability, each experiment was repeated three times, and the most stable run was reported.

This setup provides a transparent basis for comparing devices with widely different compute capabilities, memory budgets, and power envelopes, and lays the foundation for the benchmarking results reported in Section 3.2.

3.2 Benchmarking results

The performance evaluations summarized in Table 2 reveal consistent though device-dependent patterns across models, datasets, and input sizes. Under a unified protocol (batch size = 1; post-warm-up medians), we report inference time (ms), peak memory (GB), and CPU/GPU power (mW) for both 3D gesture and 2D mask-detection pipelines on AGX Orin, Jetson Nano, and RPi 3. Latency scales predictably with spatiotemporal complexity and resolution, while memory and power track model capacity; consequently, heavier inputs and larger backbones widen the gap between devices. Notably, feasibility limits appear for high-capacity models on low-end boards (e.g., N/F on RPi 3 with ResNet-50), underscoring the role of memory headroom and acceleration. For readability, Fig. 2 visualizes latency differences and Fig. 3 consolidates memory footprints; detailed numeric results remain in Table 2 for exact cross-device comparisons.

Under a unified protocol, latency is reported as the median over repeated runs after warm-up (batch size = 1, single stream). Peak memory refers to the maximum resident memory observed during inference. CPU/GPU power values are collected from on-device telemetry and averaged over runs. We additionally estimate energy per inference as

$$E \approx (P_{CPU} + P_{GPU}) \times t(1)$$

where P is power (W) and t is latency (s). This approximation enables a consistent comparison of efficiency across devices and input regimes.

Table 2. Benchmarking results of AI models across different hardware platforms. Column "Alg." indicates the algorithm type, where "GR" represents **gesture recognition** and "MD" represents **mask detection** models. "N/F" denotes cases where execution was not feasible.

Alg.	**Model**	**Data set**	**Input Size**	**Inference Time (ms)**			**Memory (G)**			**CPU (Power/mW)**			**GPU (Power/mW)**		
				AGX Orin	*Nano*	*RPi 3*	*AGX Orin*	*Nano*	*RPi 3*	*AGX Orin*	*Nano*	*RPi 3*	*AGX Orin*	*Nano*	*RPi 3*
GR	3D CNN	20BN Jester	64x64, 16F	15	50	60	2.5	1.8	0.7	360	750	850	950	370	400
			160x160, 64F	40	122	183	4.0	3.0	0.83	420	820	920	1200	550	600
	2D CNN	Jester	128x128	10	35	35	1.8	1.2	0.62	320	680	750	800	290	330
			512x512	30	102	144	3.5	2.5	0.84	380	760	840	1050	450	500
MD	ResNet50	Masked Face-Net	224x224	35	105	N/F	4.5	3.29	N/F	450	850	N/F	1150	600	N/F
			960x544	50	150	N/F	5.5	3.81	N/F	500	900	N/F	1250	650	N/F
	2D CNN	RMFD	300x300	20	60	70	2.0	1.52	0.8	340	700	780	900	350	380
			516x516	25	80	90	2.8	2.15	0.83	370	770	840	1000	420	460

3.3 Gesture recognition with 3D CNNs.

For the lighter clip (64×64, 16 F), AGX Orin achieves 15 ms processing time compared to 50 ms on Jetson Nano and 60 ms on RPi 3—approximately 3.3× and 4.0× faster than Nano and RPi 3, respectively (Table 2). When the clip size increases to 160×160, 64 F, latency scales predictably: 40 ms (Orin), 122 ms (Nano), and 183 ms (RPi 3), representing about 3.0× and 4.6× performance gains for Orin. Peak memory usage follows a similar pattern (Orin: 2.5→4.0 GB, Nano: 1.8→3.0 GB), while GPU power consumption increases with input complexity (Orin: 950→1200 mW, Nano: 370→550 mW; Table 2, Fig. 2–3).

Despite Orin's higher instantaneous GPU power draw, its energy per inference remains more efficient due to shorter processing times (approximately 14 mJ at 64×64, 16 F versus 18.5 mJ on Nano; 48 mJ at 160×160, 64 F versus 97 mJ on Nano). This demonstrates that in temporal models, superior performance translates to better energy efficiency.

3.4 Gesture recognition with 2D CNNs (Jester).

For the 128×128 resolution, the AGX Orin device demonstrates exceptional performance with an execution time of just 10 ms, while both the Jetson Nano and Raspberry Pi 3 require significantly longer processing times of 35 ms (approximately 3.5 times slower than Orin). When the resolution is increased to 512×512, the processing latency grows proportionally across all devices, reaching 30 ms for Orin, 102 ms for Nano, and 144 ms for RPi 3. As the spatial resolution is increased, there is an attendant increase in memory consumption. The range of memory consumption is from 1.8 GB (Orin), 1.2 GB (Nano), and 0.62 GB (RPi 3) at lower resolutions to 3.5 GB (Orin), 2.5 GB (Nano), and 0.84 GB (RPi 3) at higher resolutions, as detailed in Table 2 and illustrated in Figures 2–3. When compared to the three-dimensional processing configuration, the two-dimensional pipeline demonstrates inherently faster performance at comparable image dimensions; nevertheless, the computational burden of processing high-resolution images becomes particularly pronounced on the less powerful Nano and RPi 3 hardware platforms.

From an energy efficiency perspective, the two-dimensional inference workloads exhibited patterns consistent with those observed in three-dimensional processing: despite the AGX Orin's higher instantaneous power consumption, its substantially reduced execution times resulted in lower overall energy usage per inference operation (for example, approximately 8 mJ at 128×128 resolution compared to roughly 10 mJ on the Nano; similarly, about 31.5 mJ at 512×512 resolution versus approximately 45.9 mJ on the Nano), further confirming Orin's superior energy efficiency across different computational tasks..

3.5 Mask detection: 2D CNN (RMFD) and ResNet-50 (MaskedFace-Net)

The Raspberry Pi 3 cannot function (N/F) with ResNet-50 at both 224×224 and 960×544 resolutions due to memory and computing limitations (Table 2). In contrast, Orin achieves processing times of 35–50 ms (GPU power: 1150–1250 mW, memory usage: 4.5–5.5 GB), while Nano operates at 105–150 ms (GPU power: 600–650 mW, memory usage: 3.29–3.81 GB). When using the lighter 2D CNN model RMFD, all devices successfully complete inference: Orin (20–25 ms), Nano (60–80 ms), and Raspberry Pi 3 (70–90 ms). These faster speeds come with reduced memory requirements approximately 2.0–2.8 GB on Orin, 1.5–2.15 GB on Nano, and 0.8–0.83 GB on Raspberry Pi 3. These findings highlight that model capacity (such as ResNet-50) is typically the primary factor that limits performance on lower-end hardware.

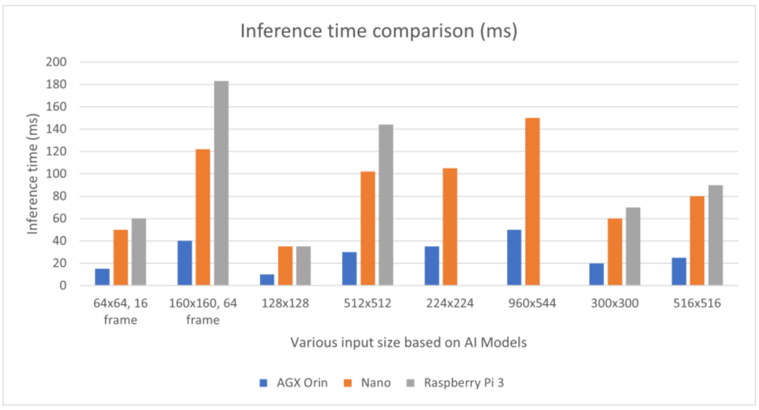

Figure 2. Inference time comparison.

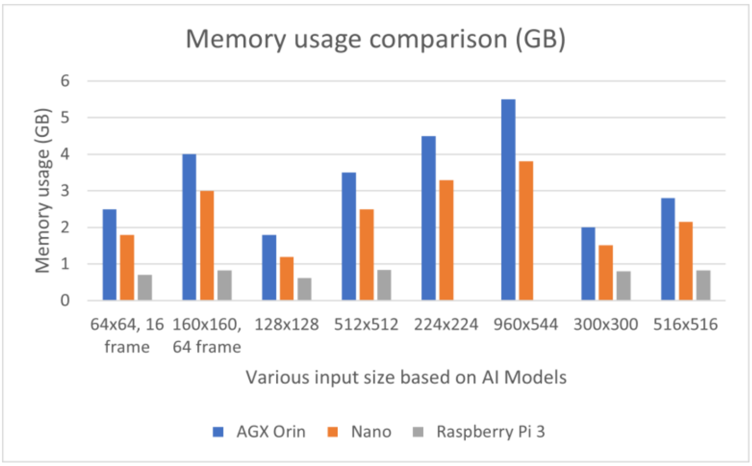

Figure 3. Memory usage comparison.

4. Discussion

The benchmarking results indicate several significant trends across devices, models, and input sizes. By examining latency, memory usage, energy efficiency, and overall device suitability, we can better understand the trade-offs that govern real-time AI deployment on edge platforms. The key observations are summarized as follows:

1. **Latency scaling.** Across all evaluated tasks, the Jetson AGX Orin consistently outperforms the Jetson Nano and Raspberry Pi 3, achieving 3–4.6× lower latency as input size and sequence length increase. This enables Orin to maintain real-time performance even with computationally demanding 3D video inputs.
2. **Memory availability.** ResNet-50 at high resolution (960×544) requires more than 5 GB of memory on Orin, whereas Nano remains below ~4 GB. The Raspberry Pi 3, with its limited memory capacity, frequently encounters infeasible (N/F) scenarios, particularly for large-scale models.
3. **Energy efficiency.** Although Orin consumes more instantaneous power, its significantly reduced inference times lead to lower overall energy per prediction compared to Nano and Raspberry Pi 3. This efficiency trend holds across both 2D and 3D tasks.
4. **Device suitability.** The AGX Orin is best suited for high-complexity applications, such as 3D CNNs or high-resolution 2D pipelines. The Jetson Nano strikes a balance between

moderate computational performance and energy efficiency, making it suitable for mid-scale deployments. Meanwhile, the Raspberry Pi 3 is most appropriate for lightweight 2D CNN models, serving as a baseline for CPU-only inference.

These findings are consistent with previous observations indicating that memory constraints and hardware acceleration paths significantly impact deployability on embedded platforms [1, 3, 4].

5. Conclusion

This study thoroughly evaluated edge computing platforms, focusing on NVIDIA's Jetson AGX Orin and Jetson Nano, in addition to the Raspberry Pi 3 as a baseline with CPU capabilities. We implemented a standardized evaluation protocol to ensure transparent and reproducible comparisons using 3D CNNs for gesture recognition and 2D CNN/ResNet-50 models for face mask detection across the 20BN-Jester, Jester, MaskedFace-Net, and RMFD datasets. Our findings indicate that the Jetson AGX Orin consistently delivers superior performance, with 3–4.6× lower latency, higher memory headroom, and enhanced energy efficiency per inference. This makes it the optimal choice for high-resolution or computationally intensive real-time applications, such as autonomous navigation, industrial AI, and healthcare diagnostics. The Jetson Nano, while less powerful, offers a balanced trade-off between efficiency and cost, ideal for moderate-scale or energy-constrained deployments. In contrast, Raspberry Pi 3 offers a valuable baseline for lightweight 2D tasks and highlights the need for optimization on CPU-only devices. These findings underscore the importance of considering factors beyond FLOPs or model size when optimizing model–device matching, including hardware-specific acceleration, thermal design, and memory constraints.

Future work will explore model compression techniques, such as pruning, quantization, and distillation. It will also integrate with next-generation accelerators, including Jetson Thor and ASICs. Additionally, it will evaluate additional edge AI tasks, such as tracking and anomaly detection. Finally, it will explore adaptive and federated learning for personalization across heterogeneous environments.

Overall, this study provides actionable guidelines for balancing accuracy, speed, and energy efficiency in edge AI deployments, contributing to sustainable and reliable real-time intelligence on resource-limited platforms.

REFERENCES

1. Süzen, A.A., Duman, B., Şen, B.: Benchmark Analysis of Jetson TX2, Jetson Nano and Raspberry Pi using Deep CNN. In: Proc. Int. Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Ankara, Turkey, 26–28 June 2020, pp. 1–5 (2020).
2. George, A., Ecabert, C., Otroshi-Shahreza, H., Kotwal, K., Marcel, S.: EdgeFace: Efficient Face Recognition Model for Edge Devices. IEEE Trans. Biometrics, Behavior, and Identity Science 6(2), 158–168 (2024).
3. Ullah, S., Kim, D.-H.: Benchmarking Jetson Platform for 3D Point Cloud and Hyperspectral Image Classification. In: 2020 IEEE Int. Conf. on Big Data and Smart Computing (BigComp), pp. 477–482 (2020).
4. Choe, M., Lee, S., Sung, N.M., Jung, S., Choe, C.: Benchmark Analysis of Deep Learning-based 3D Object Detectors on NVIDIA Jetson Platforms. In: Int. Conf. on ICT Convergence (ICTC), pp. 10–12 (2021).
5. Jovović, I., Babić, D., Čakić, S., Popović, T., Krčo, S., Knežević, P.: Face Mask Detection Based on Machine Learning and Edge Computing. In: 21st Int. Symp. INFOTEH-JAHORINA (INFOTEH), East Sarajevo, Bosnia and Herzegovina, pp. 1–4 (2022). https://doi.org/10.1109/INFOTEH53737.2022.9751311
6. Zhang, Z., Zhang, X., Peng, C., Xue, X., Sun, J.: Efficient and Accurate Approximations of Nonlinear Convolutional Networks. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1984–1992 (2016).

7. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016).
8. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C.: MobileNetV2: Inverted Residuals and Linear Bottlenecks. In: Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 4510–4520 (2018).
9. Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q.V., Adam, H.: Searching for MobileNetV3. In: Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV), pp. 1314–1324 (2019).
10. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning Spatiotemporal Features with 3D Convolutional Networks. In: Proc. IEEE Int. Conf. on Computer Vision (ICCV), pp. 4489–4497 (2015).
11. Carreira, J., Zisserman, A.: Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In: Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 4724–4733 (2017).
12. Materzyńska, J., Berger, G., Bax, I., Memisevic, R.: The Jester Dataset: A Large-Scale Video Dataset of Human Gestures. In: Proc. IEEE/CVF Int. Conf. on Computer Vision Workshops (ICCVW), Oct (2019).
13. Cabani, A., Hammoudi, K., Benhabiles, H., Melkemi, M.: MaskedFace-Net: A Dataset of Correctly/Incorrectly Masked Face Images in the Wild. Data in Brief 35 (2021).
14. Wang, Z., Wang, G., Huang, B., Xiong, Z., Hong, Q., Wu, H., Yi, P., Jiang, K., Wang, N., Pei, Y., Chen, H., Miao, Y., Huang, Z., Liang, J.: Masked Face Recognition Dataset and Application. arXiv:2003.09093 (2020).
15. Ignatov, A., Timofte, R., Chou, W., Wang, K., Wu, M., Hartley, T., Van Gool, L.: AI Benchmark: Running Deep Neural Networks on Android Smartphones. In: Computer Vision – ECCV 2018 Workshops, LNCS, pp. 288–314 (2019).
16. NVIDIA Corporation: Jetson AGX Orin — Product page. Online at: https://developer.nvidia.com/embedded/jetson-agx-orin (accessed 22 Aug 2025).
17. NVIDIA Corporation: Jetson Nano Developer Kit — Product page. Online at: https://developer.nvidia.com/embedded/jetson-nano-developer-kit (accessed 22 Aug 2025).