

Hybrid vision-language models and on-device object detection for Mobile Augmented Reality Learning using the NARV algorithm

A. Rahman

Abstract—Mobile Augmented Reality (MAR) is increasingly employed to deliver immersive instructional content in underserved classrooms, with object recognition typically offloaded to cloud-based vision-language models. The challenge encountered in such deployments diverges fundamentally from conventional assumptions. Empirical field measurements reported in this article demonstrate that the binding constraint is not elevated latency, but rather intermittent cloud service unreachability, rendering strategies built around extending timeouts or deferring transmissions until connection recovery of no benefit to students standing directly before physical objects awaiting an immediate answer. Existing reliability mechanisms recover passively through local caching and manual user intervention, steer computation based on input content complexity under simulated bandwidth throttling or achieve network independence only by necessitating interconnected edge device clusters. This article proposes Network-State Adaptive Resilient Vision (NARV), a pipeline that combines bounded retry orchestration governed by linear backoff with a reactive substitution pathway to an on-device detector running continuously throughout the scanning phase, ensuring that substitute label sets are pre-accumulated and readily available the moment all cloud attempts are exhausted without incurring additional latency. Quantitative in-the-wild evaluations were conducted across 90 sessions from 90 unique participants on three heterogeneous Android hardware tiers in Sumbawa Besar, Indonesia. The pipeline achieved a continuity rate of 100.0%, with 84.4% of sessions resolved via the cloud pathway and 15.6% via on-device substitution. Out of 49 failed cloud attempts, 47 failed instantaneously with a median of 29 ms rather than reaching the timeout boundary, 30 out of 35 backoff schedulings executed within a 0.038-second deviation from the model, and memory growth remained bounded at 11.86 MB per session.

Keywords—network resilience; mobile augmented reality; hybrid edge-cloud inference; vision-language models; in-the-wild evaluation

I. INTRODUCTION

MOBILE Augmented Reality (MAR) has emerged as a promising pedagogical paradigm to bridge global educational divides by enhancing spatial abilities, motivation, and cognitive engagement beyond physical classrooms [1], [2], while cultivating social and physical presence unattainable through conventional laptop- or whiteboard-based instruction [3]. Empirically, a quasi-experimental study on 173 primary school students in urban and rural South China proved that a dual-scaffolding mobile AR approach significantly improves

achievement, higher-order cognition, and self-efficacy, thereby narrowing urban-rural disparities ($F = 12.23$, $p < 0.01$) [2]. Nonetheless, a systematic review of 55 studies across 25 developing countries revealed a persistent scarcity of consistent high-quality evidence supporting educational technology in these regions [4]. This challenge is aggravated by uneven infrastructure and high data costs for low-income learners [5], as well as the hardware heterogeneity inherent in bring-your-own-device deployments. Indeed, an in-the-wild city-scale study involving 439 students across 118 group sessions demonstrated that connection losses and device discrepancies frequently disrupt learning trails, establishing network-independent execution as a fundamental feasibility prerequisite rather than an optional feature [6]. Consequently, an urgent architectural demand exists to shift computational and object recognition workloads to the local client device, guaranteeing reliable instructional delivery, bounded latency, and operational feasibility on low-specification smartphones under severe network constraints.

Beyond their generative capabilities, large language models (LLMs) and large vision-language models (LVLMs) harbor two distinct vulnerabilities: an internal propensity for factual hallucinations compounded by visual-textual discrepancies [7], [8], and an external reliance on network connectivity where cloud-dependent inference renders any connection outage a total service failure. In Indonesia, this external vulnerability is exacerbated by severe infrastructure disparities, with 1,530 villages entirely lacking telecommunication signals [9], [10]. Even within coverage zones, a PRISMA-based systematic review of 142 articles confirmed that rural environments are plagued by intermittent backhaul, sparse gateway access, and harsh propagation conditions, noting that urban-centric literature largely overlooks these realities [11]. Consequently, connectivity-reliant systems risk severe performance degradation, as evidenced by conventional distributed protocols that suffer extreme latency spikes and collapse during network disruptions [12]. Although the common store-and-retransmit strategy mitigates transmission failures in asynchronous monitoring [11], it is inapplicable to interactive learning sessions where students directly face physical objects and expect instantaneous feedback, while repeated service disruptions erode user trust and undermine long-term sustainability in marginalized areas [13]. This intersection of cloud VLM dependency and network intermittency creates an urgent demand for an architecture that guarantees immediate

Author is with Warsaw University of Technology, Poland (e-mail: a.rahman.dokt@pw.edu.pl).



© The Author(s). This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY 4.0, <https://creativecommons.org/licenses/by/4.0/>), which permits use, distribution, and reproduction in any medium, provided that the Article is properly cited.

output availability upon request rather than deferred delivery upon connection recovery.

To address these reliability challenges, recent research has explored integrating generative AI into education, although the lack of output trustworthiness and reliability remains a widely recognized ethical obstacle [14]. Concurrently, cloud-edge computing architectures offer an effective paradigm for developing regions, exemplified by deep-learning disease diagnosis frameworks that partition computational loads between edge and cloud devices to preserve high accuracy over constrained infrastructure [15]. Similarly, offline-first architectures in mobile reporting establish local execution as the default operational baseline rather than an exception [16]; however, while their local-first principle is foundational, their deferred synchronization models cannot support real-time inference where outputs are expected instantaneously. In parallel, advances in lightweight deep learning, spanning efficient neural network design, model compression, and hardware acceleration, enable models to be deployed directly onto resource-constrained mobile devices without complete cloud reliance [17]. Together, these three paradigms, encompassing cloud-edge computing, offline-first design, and lightweight deep learning, underscore the urgent need for an integrated architecture specifically engineered to guarantee reliable AI-based object recognition under volatile network conditions by leveraging on-device capabilities as an autonomous fallback when cloud connectivity is severed.

Although augmented reality has demonstrated pedagogical efficacy, validations are predominantly conducted in laboratories or over stable networks where network quality is neglected as a research variable [18]. In contrast, a multi-country field study revealed that direct in-situ trials uncover severe contextual constraints, such as sluggish connectivity and scarce resources, that never emerge during controlled development [19]. Similarly, a survey of 281 publications on sixth-generation edge AI inference concluded that existing optimizations are largely validated within simulations that deviate from physical realities, establishing the simulation-to-reality divide as an open, formidable challenge [20]. This discrepancy between idealized evaluations and operational field realities necessitates a distinct approach. To address this gap, this study presents three core scientific contributions. First, an orchestration architecture for cloud vision-language model invocations equipped with retry, timeout, and staged linear backoff, guaranteeing service reliability under volatile networks while enforcing a calculable latency upper bound that is never exceeded. Second, a hybrid fallback mechanism integrating cloud Gemini Vision with on-device YOLO, preserving continuous object recognition even when connectivity to cloud services is severed. Third, an authentic real-world field deployment in rural Sumbawa Besar, Indonesia, measuring computational resource footprints across three student-owned hardware tiers to provide empirical validation rarely captured in laboratory-bound literature.

II. RELATED WORK

A. Mobile Augmented Reality and Object Recognition in Underserved Pedagogical Environments

A deep-learning-based augmented reality (AR) framework combining an automated synthetic image generation pipeline

with a YOLOv7-based object detector was proposed by Kaviyaraj and Mohan [21], validated through an impact evaluation involving 210 elementary school students and achieving an average detection accuracy of 97.2% with a real-time inference speed of 45 FPS on classroom tablets. To address the rigidity of static learning materials, an AR Mobile Learning System (ARMLS) utilizing marker-based tracking via AR cards and QR code triggers was proposed by Tarnng et al. [22], demonstrating a significant reduction in cognitive load ($M = 12.94$ vs. $M = 16.38$, $p = 0.038$) and targeted learning achievement gains for low- to medium-performing students, as confirmed through Johnson-Neyman analysis. Furthermore, a two-phase detect-and-describe (D&D) framework integrating YOLOv8 for object extraction with a VGG16-LSTM encoder-decoder for automated semantic description generation was developed by Lv and Yao [23], achieving an object recognition accuracy of 94.4% and a description accuracy of 96.5% on a self-constructed preschool education dataset.

The limitation of the literature is that existing mobile AR object recognition architectures for education either rely on deep learning detectors executed entirely on-device without addressing cloud connectivity dependencies, depend on rigid, low-interactivity marker-based tracking that cannot generalize to open real-world objects beyond pre-registered cards or QR codes, or remain validated exclusively within offline research environments without ever being genuinely deployed as a real-time mobile AR application under authentic field network conditions.

B. Cloud Vision-Language Models and Object Recognition on Mobile and Edge Devices

A comprehensive survey on edge vision-language model (VLM) deployment by Sharshar et al. [24] reported that pairing lightweight detectors like YOLOv7 with VLMs facilitates real-time detection and scene description in resource-constrained environments, reducing latency and dependence on stable connections by processing data closer to the edge. Concurrently, a survey of 268 papers by Li et al. [25] mapped collaborative edge SLM and cloud LLM execution into task assignment, splitting, and hybrid strategies with recovery mechanisms such as local re-execution, but concluded that existing fallback and switching triggers rely on ad hoc confidence thresholds, leaving resilience under unstable connections an open challenge. Empirically evaluating GPT-4-Vision and Gemini Pro Vision in safety-critical domains, Ashqar et al. [26] achieved up to 90% true-positive vehicle detection and 74.67% critical driving event accuracy, while identifying object hallucinations and computational bottlenecks as core constraints. Furthermore, Huang et al. [27] developed Vinci, a wearable egocentric VLM assistant offloading inference to the cloud that achieved sub-second latency and 91.0% chat accuracy under optimal conditions, yet explicitly acknowledged that "performance may degrade or even fail completely in low-bandwidth or high-latency environments." Conversely, Kalam et al. [28] proposed a fully on-device coarse-to-fine pipeline combining YOLO, segmentation, and classification to enable fine-grained detection without internet connectivity, albeit sacrificing the high-level reasoning capabilities of cloud VLMs in exchange for network independence.

The limitation of the literature is that existing VLM-based edge object recognition systems either remain confined to

architectural surveys lacking concrete failure recovery mechanisms, rely on ad hoc confidence triggers and upward offloading that collapses when cloud services become unreachable, validate detection accuracy exclusively under stable connectivity while ignoring network volatility, deploy cloud-dependent physical systems that acknowledge total failure under degraded networks yet leave this vulnerability unresolved, or abandon cloud reasoning entirely to ensure offline execution at the expense of fine-grained accuracy.

C. Reliability, Fault Tolerance, and Hybrid Device-Cloud Inference for Object Detection

To preserve service continuity under network disruptions, a modular smart agriculture application with Dynamic Feature Modules developed by Syaifudin et al. [29] employed exponential backoff retries, local caching, and fallback notifications; however, this recovery path remained passive, requiring manual retries or extension consultations rather than restoring functionality autonomously. To minimize transmission overhead, the content-aware device-cloud collaborative inference (DCCI) framework proposed by Hu et al. [30] utilized a hard-case discriminator to route visually complex inputs to the cloud while processing simple inputs locally, achieving over 90% on-device memory reduction and a 1.2-fold latency improvement over state-of-the-art DNN partitioning, although offloading was governed by input complexity and simulated bandwidth constraints on a controlled WLAN rather than genuine connection failures. Furthermore, Zou et al. [31] proposed LC-CPDI, an edge-to-edge framework for surveillance person detection that cut inference delay by up to 41.8% via logical correction and selective skipping across edge clusters, yet it relied on interconnected multi-node edge infrastructure evaluated over a 1000 Mbit/s wired connection

assumed not to be a bottleneck, an architecture inaccessible to a single student-owned smartphone in an underserved classroom.

The limitation of the literature is that existing reliability mechanisms for cloud-dependent AI inference either recover from connectivity failures through passive caching and manual user intervention rather than autonomous full functional restoration, partition computation between device and cloud based on input content complexity and simulated bandwidth constraints rather than real-time authentic connection failures, or achieve network independence only by requiring interconnected edge device clusters that are unavailable in single-device, resource-constrained educational deployments.

III. RESULTS AND DISCUSSIONS

To evaluate the performance and robustness of the proposed Network-State Adaptive Resilient Vision (NARV) pipeline, extensive field trials were conducted in-the-wild. The testing environment was situated in Sumbawa Besar, West Nusa Tenggara, Indonesia, representing a rural and remote educational setting characterized by constrained network infrastructures.

The evaluated system is the educational AR application RoboAR, built upon the Unity game engine, featuring a cloud-based object identification path via Gemini Vision and an on-device substitution path based on a YOLO model executed directly on the device. The evaluation was performed across three distinct hardware tiers to assess the scalability of the algorithm. The specifications and classification of the experimental mobile devices are structured and presented in Table I.

TABLE I
SPECIFICATIONS AND CLASSIFICATION OF EXPERIMENTAL DEVICES

Device Model	Hardware Tier	Chipset	RAM Capacity	GPU	OS Version	Sessions (n)
Samsung SM-A055F	Entry-Level	MediaTek Helio G85	4 GB (3,678 MB)	Mali-G52 MC2 (1,024 MB)	Android 14 / API-34	30
Infinix X1102 (Xpad 20)	Entry-Mid	MediaTek Helio G88	8 GB (7,703 MB)	Mali-G52 MC2 (2,048 MB)	Android 15 / API-35	30
Xiaomi 24117RN76E	Mid-Range	MediaTek Helio G99-Ultra	8 GB+ (7,655 MB)	Mali-G57 MC2 (2,048 MB)	Android 15 / API-35	30

For verification, each device executed thirty (30) continuous learning sessions conducted by unique and non-overlapping participants across devices, yielding a total of $N = 90$ valid sessions from 90 participants. These experimental trials generated 541 session event rows and 9,226 detailed telemetry rows encompassing latency metrics, memory consumption, frame rate, computational cost, and navigation stability. The mean duration of a single session was recorded at 30.9 seconds with a standard deviation (SD) of 7.3 seconds and a range of 17 to 59 seconds, whereas the mean time until the first object identification result was obtained stood at 25.0 seconds ($SD = 6.4$). Throughout this section, M denotes the mean, SD denotes the standard deviation, and N denotes the number of observations in the group under discussion. The network environment under which the trials were executed was intermittent and unreliable, with cloud service availability

fluctuating throughout the testing period, whose quantitative characterization is detailed in Subsection III-A.

A. Experiment 1: Orchestration Latency and Retry Behavior Analysis

The first stage of evaluation analyzed the latency performance of the proposed NARV pipeline. Unlike conventional cloud-based architectures that model response time as a single isolated call, NARV executes a sequence of attempts with backoff intervals in between. The total orchestration time actually elapsed in a single session is therefore decomposed into the accumulated latency of all executed attempts and the accumulated backoff intervals between those attempts, as expressed in (1):

$$T_{\text{session}} = \sum_{i=1}^a T_{\text{attempt},i} + \sum_{n=1}^{a-1} B(n) \quad (1)$$

where $T_{session}$ represents the total orchestration time elapsed in a single session. The parameter a denotes the number of attempts actually utilized in that session, with $1 \leq a \leq N_{max}$. Here, $T_{attempt,i}$ represents the latency of the i -th attempt, defined as the time interval between sending the request and receiving either a response or a failure. $B(n)$ denotes the linear backoff interval scheduled after the n -th attempt fails, which grows proportionally to the attempt index according to $B(n) = B_{base} \cdot n$, with $B_{base} = 1.5$ s serving as the baseline backoff delay. $N_{max} = 3$ is the maximum number of attempts permitted by the orchestrator, and $T_{timeout} = 10$ s denotes the timeout threshold enforced on each attempt.

As shown in Fig. 1a, the latency distribution of successful cloud calls differed substantially across hardware tiers. The median latency was recorded at 2,414 ms for the Xiaomi ($M = 2,497.9$, $SD = 485.6$, $N = 27$), 4,311 ms for the Infinix ($M = 4,500.2$, $SD = 1,051.7$, $N = 30$), and 3,004 ms for the Samsung ($M = 3,036.5$, $SD = 688.5$, $N = 19$). In aggregate, the latency of successful calls reached $M = 3,422.9$ ms ($SD = 1,197.2$, median 3,135 ms, range 1,823 to 7,922 ms, $N = 76$). All medians remained below the PASS threshold of 8,000 ms, yet none achieved the EXCELLENT threshold of 2,000 ms. This finding confirms that the cloud path at the testing location operates within a viable yet far from ideal margin, such that full dependence upon it cannot be technically justified.

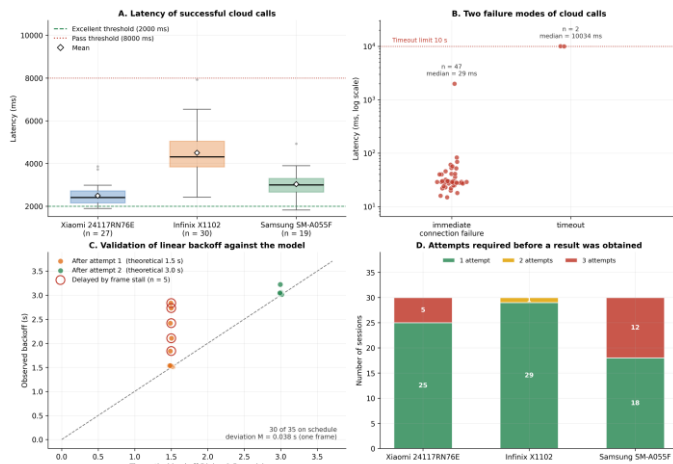


Fig. 1. Orchestration latency and retry behavior of the NARV pipeline across three device tiers.

There is a paradox that warrants explanation. The Infinix device (Entry-Mid) recorded the highest latency, yet was simultaneously the only device that never required on-device substitution. Conversely, the Samsung (Entry-Level) yielded lower latency when its calls succeeded, but lost connectivity most frequently. This pattern demonstrates that connection speed and connection reliability are two distinct dimensions, and a system optimized solely for latency would overlook the failure mode that is actually dominant in the field.

Fig. 1b decomposes the 49 failed attempts into two starkly contrasting modes. A total of 47 attempts (95.9%) failed instantaneously with a median of 29 ms, largely comprising DNS resolution failures, whereas only 2 attempts (4.1%) actually reached the $T_{timeout}$ boundary with a median of 10,034

ms. Cross-verification revealed perfect concordance between these 49 failed attempts and the 49 Vision_API_Error events logged in the session telemetry. This carries significant practical implications. Because failed attempts returned within tens of milliseconds, the empirical time cost of a three-attempt chain in the field amounted to only $M = 5.71$ s, far below the theoretical upper bound of 34.5 seconds implied by the orchestrator parameters. In other words, the timeout parameter was virtually never binding, and the true cost of the retry mechanism was dominated by backoff intervals rather than connection wait times.

Fig. 1c validates the adherence of backoff scheduling to the linear model $B(n) = 1.5n$, where 30 out of 35 schedulings (85.7%) executed punctually with a mean deviation of merely 0.038 seconds ($SD = 0.037$) relative to theoretical benchmarks. Observed delays ranged from 1.513 to 1.540 seconds for $B(1) = 1.5$ s and from 3.016 to 3.062 seconds for $B(2) = 3.0$ s, with this single-frame deviation directly resulting from Unity's WaitForSeconds resolving at subsequent frame boundaries. The remaining five schedulings suffered delays of up to 1.335 seconds, with four occurring on the Entry-Level device. Telemetry confirmed that these delayed sessions exhibited an average minimum frame rate of 1.66 FPS, compared to 4.37 FPS in punctual sessions. These bottlenecks stemmed from device render loop congestion rather than scheduler faults, demonstrating that coroutine timing precision is ultimately bounded by the device rendering budget.

As illustrated in Fig. 1d, the distribution of required attempts exhibits a bimodal pattern: 72 sessions concluded on the first attempt, only 1 on the second, and 17 required all three attempts, averaging 1.39 attempts out of N_{max} with 18 sessions (20.0%) requiring retries. This bimodal profile indicates that connectivity failures were session-persistent rather than transient, as failed initial attempts almost invariably led to subsequent failures, thereby challenging the conventional assumption that network disruptions are momentary and mutually independent. Among the 18 retried sessions, 4 sessions (22.2%) successfully recovered to the cloud pathway, while the resolution of the remaining 14 sessions is examined in Subsection III-B.

To quantify network volatility throughout the experimental trials, jitter is calculated as the absolute variance between the total orchestration times of two consecutive sessions, as expressed in (2):

$$J = |T_{session,k} - T_{session,k-1}| \quad (2)$$

where J denotes jitter, representing the measure of inter-session network condition volatility. $T_{session,k}$ is the total orchestration time in the k -th session according to chronological execution order, and $T_{session,k-1}$ is the total orchestration time in the preceding session. The jitter analysis confirmed substantial instability, with $M = 1,695.9$ ms ($SD = 2,169.1$, range 5 to 11,186 ms, $N = 89$). Fluctuations of this magnitude imply that two consecutive sessions executed at the same physical location can encounter entirely disparate network conditions, rendering adaptation strategies that rely on prior network measurements as predictive baselines unreliable. This justification underpins the architectural design of NARV as a reactive substitution mechanism rather than proactive path selection.

B. Experiment 2: Validation of Service Continuity and Substitution Behavior

The core objective of the NARV pipeline is to guarantee that every learning session consistently yields object identification, regardless of connectivity availability while the session is underway. The second stage of evaluation therefore measures system reliability at the session level through three complementary metrics, as expressed in (3):

$$R_{cont} = R_{cloud} + R_{sub} = \frac{S_{cloud} + S_{sub}}{S_{total}} \times 100\% \quad (3)$$

where R_{cont} is the Continuity Rate, defined as the proportion of sessions that successfully yield object identification through either path. R_{cloud} represents the Cloud Success Rate, denoting the proportion of sessions completed entirely via the cloud path without requiring substitution. R_{sub} denotes the Substitution Rate, which is the proportion of sessions resolved via the on-device YOLO model after all cloud attempts are exhausted. S_{cloud} represents the number of sessions completed through the cloud path, S_{sub} is the number of sessions resolved through the substitution path, and S_{total} denotes the total number of analyzed sessions.

The formulation in (3) demonstrates the structural characteristic of the NARV pipeline, namely that R_{cont} is the direct summation of both completion pathways. There exists no third category of sessions failing to produce any output, insofar as either of the two pathways can be completed.

1. Service Continuity Under Network Failures

The quantitative results across the 90 sessions are summarized in Fig. 2b. The NARV pipeline achieved $R_{cont} = 100.0\%$, meaning that all 90 sessions successfully produced object identification without a single session terminating in service failure. Of this total, 76 sessions were resolved via the cloud pathway ($R_{cloud} = 84.4\%$) and 14 sessions were completed through on-device substitution ($R_{sub} = 15.6\%$).

As broken down in Fig. 2a, the substitution workload was distributed highly unevenly across hardware tiers. While the Infinix (Entry-Mid) completed all 30 sessions via the cloud without substitution, the Xiaomi (Mid-Range) required substitution in 3 out of 30 sessions, and the Samsung (Entry-Level) utilized it in 11 out of 30 sessions, exceeding one-third of its trials. This confirms that substitution is not an idle fallback, but a vital mechanism bearing substantial operational loads under constrained conditions. Crucially, despite this disparity, the total bar height across all devices remains identical at 30 in Fig. 2a, demonstrating uniform service continuity across all tiers.

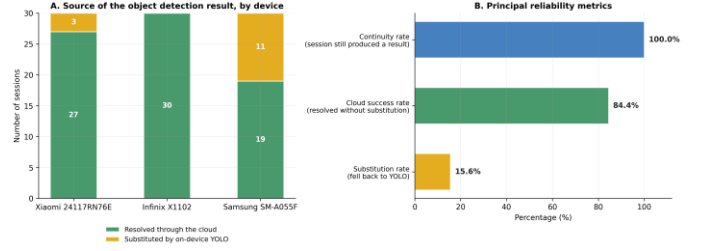


Fig. 2. Reliability of the NARV pipeline under field network conditions.

Correlating these results with the retry behavior in Subsection III-A reveals a distinct division of roles: of the 18 sessions failing on the first attempt, 4 sessions (22.2%) recovered to the cloud via retry, while 14 sessions (77.8%) were resolved through on-device substitution. These mechanisms perform complementary functions: substitution guarantees service continuity by preventing the 14 failed sessions from terminating without output, whereas retry safeguards output quality by preventing the 4 recovered sessions from degrading to lower-granularity local detection. Although this 22.2% to 77.8% ratio reflects the persistent DNS-dominated failures of the testing site, environments with transient congestion could see a higher retry contribution. Because pathway proportions cannot be determined a priori and depend on dynamic field realities, maintaining both mechanisms concurrently in a hybrid architecture is thoroughly justified.

2. Result Granularity on the Substitution Pathway

As summarized in Table II, this continuity entails a trade-off in semantic richness between the two resolution pathways. Across 76 sessions, the cloud pathway produced specific, contextual labels drawn from a vocabulary of 49 distinct classes, such as *Perfume bottle*, *Candy wrapper*, and *Figurine*. Conversely, the substitution pathway was restricted to 10 broad on-device categories dominated by *Laptop* and *Cell Phone*, with occasional out-of-context misclassifications such as *Toilet*, *Motorcycle*, and *Vehicle*, each occurring once.

This disparity contextualizes NARV's architectural contribution: the substitution pathway is not designed to achieve semantic parity with the cloud, but rather to ensure session continuity with coarser, immediately available results rather than halting instructional delivery entirely. In resource-constrained educational settings where service failure directly disrupts student engagement, this trade-off between granularity and continuity is highly favorable. Consequently, NARV positions retry to preserve high-fidelity results for as long as feasible, and substitution to guarantee that sessions never terminate without output.

TABLE II
COMPARISON OF IDENTIFICATION RESULT GRANULARITY ACROSS PATHWAYS

Resolution Pathway	Sessions (N)	Object Per Session	Median	Unique Label Count
Samsung SM-A055F	Entry-Level	MediaTek Helio G85	4 GB (3,678 MB)	Mali-G52 MC2 (1,024 MB)
Infinix X1102 (Xpad 20)	Entry-Mid	MediaTek Helio G88	8 GB (7,703 MB)	Mali-G52 MC2 (2,048 MB)
Xiaomi 24117RN76E	Mid-Range	MediaTek Helio G99-Ultra	8 GB+ (7,655 MB)	Mali-G57 MC2 (2,048 MB)

C. Experiment 3: Computational Resource Footprint Analysis

For educational AR applications deployed on student-owned devices, memory efficiency constitutes a fundamental feasibility prerequisite. Devices with constrained memory capacities execute operating system memory management mechanisms that forcibly terminate applications if their memory footprint expands uncontrollably, thereby disrupting learning sessions not due to network failures, but as a direct result of on-device limitations. The third stage of evaluation therefore measures the memory footprint and rendering fluidity of the NARV pipeline across all three hardware tiers.

The memory growth within a single session is calculated as the difference between the peak memory consumption and the memory usage at the onset of the scanning phase, as expressed in (4):

$$\Delta M_s = M_{peak,s} - M_{init,s} \quad (4)$$

where ΔM_s denotes the memory growth in the s -th session. $M_{peak,s}$ represents the peak memory consumption recorded throughout that session, and $M_{init,s}$ is the memory consumption at the initial sampling point after the participant enters the session. It should be noted that $M_{init,s}$ does not represent the idle state of the application, but rather the initial state of the scanning phase, as telemetry sampling commences only after the session is actively underway.

As illustrated in Fig. 3a, the memory footprint of the NARV pipeline is low and exceptionally stable across all hardware tiers. The breakdown per device is as follows:

- The Infinix device (Entry-Mid) recorded the lowest memory consumption, with a mean peak of 232.1 MB and an absolute highest peak of 242 MB, accompanied by a very tight distribution.
- The Xiaomi device (Mid-Range) recorded a mean peak of 261.0 MB with an absolute highest peak of 265 MB, likewise exhibiting a tight distribution.
- The Samsung device (Entry-Level) recorded a mean peak of 273.3 MB with an absolute highest peak of 294 MB, accompanied by the widest distribution among the three.

Fig. 3b presents the memory growth per session calculated according to (4). In aggregate, memory growth across a single session amounted to merely $M = 11.86$ MB ($SD = 9.93$), with averages of 6.00 MB on the Xiaomi, 9.27 MB on the Infinix, and 20.30 MB on the Samsung. These minute and bounded values demonstrate that the NARV pipeline experiences no uncontrolled memory accumulation across the session life cycle, rendering the risk of forced termination by the operating system negligible across all tested devices.

Nonetheless, interpreting absolute values alone can be misleading, given that the three devices possess vastly different memory capacities. Proportional memory utilization is therefore calculated as the ratio between peak consumption and the device memory capacity, as expressed in (5):

$$U_{mem} = \frac{M_{peak}}{M_{device}} \times 100\% \quad (5)$$

where U_{mem} represents the proportional memory utilization, M_{peak} denotes the mean peak memory consumption on the respective device, and M_{device} is the physical memory capacity of that device as listed in Table I.

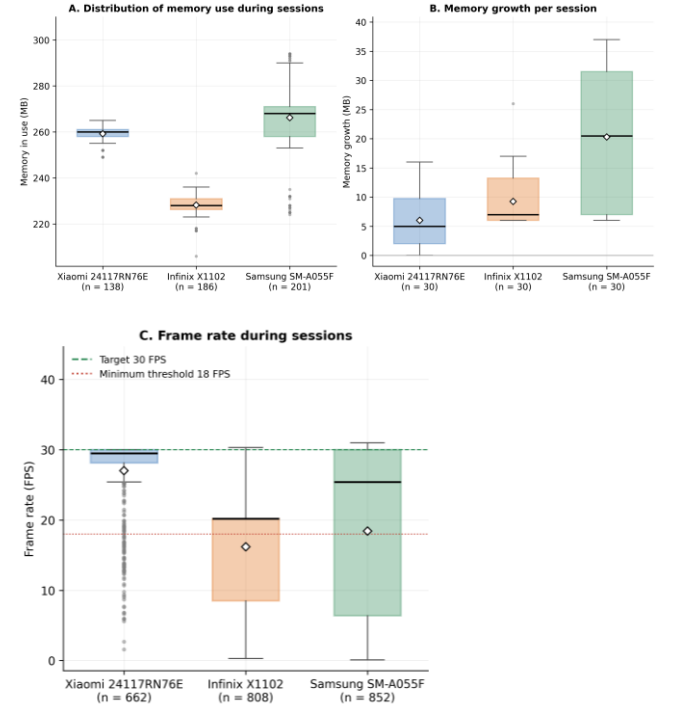


Fig. 3. NARV pipeline resource load on heterogeneous devices.

Calculations according to (5) yield a U_{mem} of 3.01% on the Infinix, 3.41% on the Xiaomi, and 7.43% on the Samsung. Consequently, the actual memory pressure experienced by the Entry-Level device is more than double that of the other two devices, even though the difference in absolute consumption appears minor. This finding concurrently explains why this device exhibited the widest memory dispersion in Fig. 3a and the largest memory growth in Fig. 3b.

Fig. 3c presents the frame rate distribution throughout the sessions, where the system limitations become most apparent. The Xiaomi device maintained favorable rendering fluidity with a median of 29.5 FPS ($M = 27.02$, $SD = 5.69$, $N = 662$), with only 11.5% of its telemetry samples falling below the minimum threshold of 18 FPS. Conversely, the Infinix device recorded a median of 20.2 FPS ($M = 16.20$, $SD = 7.56$, $N = 808$) and the Samsung device recorded a median of 25.4 FPS ($M = 18.44$, $SD = 11.77$, $N = 852$), with 46.8% and 47.3% of their samples falling below this threshold, respectively. In other words, nearly half of the session duration across both lower-tier devices occurred below the established fluidity threshold.

These rendering bottlenecks are not merely a matter of visual comfort, but carry measurable consequences for orchestrator behavior. As reported in Subsection III-A, the five backoff schedulings that deviated from the model occurred during sessions with an average minimum frame rate of 1.66 FPS, four of which were on the Entry-Level device. This demonstrates that device rendering budget

constraints propagate into scheduling precision limitations, confirming that both findings stem from the same root cause.

Overall, the results of Experiment 3 demonstrate that the NARV pipeline is viable for execution on lower-end devices typically owned by students in rural regions, maintaining a lightweight, stable memory footprint without risking forced application termination. Nonetheless, rendering fluidity on Entry-Level and Entry-Mid devices has not yet reached an optimal standard, representing an operational limitation to be addressed through further optimization in subsequent research.

IV. MATHEMATICAL FORMULATION AND ALGORITHM DESIGN

The core architecture of the NARV pipeline is designed to address two primary sources of reliability failures in AI-based visual tutoring systems within unstable network environments, namely transient failures or elevated latency during cloud API invocations, and total network connectivity failures that render cloud services entirely inaccessible. This section establishes the formal mathematical model of the adaptive retry orchestration mechanism, the reactive object detection substitution mechanism, and the comprehensive system execution algorithm. All orchestrator parameters, namely N_{max} , $T_{timeout}$, B_{base} , and the backoff function $B(n)$, have been defined and empirically validated in Subsection III-A.

A. Orchestration of Retry, Timeout, and Staged Backoff

Let a request to the cloud service be represented as a call operation C , with each invocation attempt bounded by a timeout threshold $T_{timeout}$. Given n as the attempt index, where $n = 1, 2, \dots, N_{max}$, the success of a single attempt is not determined solely by successful network transmission, but rather by the fulfillment of all necessary criteria ensuring the results can be presented to the student, as expressed in (6):

$$\sigma(n) = \begin{cases} 1, & \text{if } R_n \text{ is received within } T_{timeout}, R_n \text{ is parseable, and } |L_n| \geq 1 \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

where $\sigma(n)$ represents the success indicator of the n -th attempt, R_n denotes the response received from the cloud service on that attempt, and L_n is the set of object labels successfully extracted from R_n . The formulation in (6) establishes that a response received within the deadline but containing an empty payload or failing to parse is treated as equivalent to a network failure, because all three equally fail to produce usable output. This classification forms the basis for recording events in the telemetry data.

When $\sigma(n) = 0$ and n has not yet reached N_{max} , the orchestrator waits for the backoff interval $B(n)$ before reattempting. Unlike the exponential schemes commonly deployed in distributed systems, this delay increases incrementally and linearly with respect to the index of the attempt that has just failed. There exists neither an upper bound clamp nor a randomized jitter component applied to this interval, rendering its growth purely proportional to n .

The selection of this linear scheme is motivated by the specific failure characteristics of the target environment. Exponential backoff is engineered to alleviate contention across saturated resources, making delay doubling rational

when failures stem from system overload. In this experimental environment, failures were predominantly driven by service unreachability as reported in Subsection III-A, which does not recover any faster by extending delays exponentially. Linear growth was therefore chosen to maintain a concise and calculable total waiting time, considering that students are awaiting results in real time as the process unfolds. Under the configuration deployed in the RoboAR implementation, the scheduled backoffs are 1.5 seconds following the first failed attempt and 3 seconds following the second failed attempt, prior to the execution of the third and final attempt.

The retry loop terminates upon meeting either of two conditions, whichever occurs first, as expressed in (7).

$$\text{terminate} \Leftrightarrow (\exists n \leq N_{max} \mid \sigma(n) = 1) \vee (n = N_{max}) \quad (7)$$

The formulation in (7) confirms that orchestration termination is purely governed by the success of an attempt or the exhaustion of the maximum attempt threshold, without a separate cumulative time budget. Nonetheless, because both the number of attempts and the intermediate intervals are bounded, the total orchestration time maintains a calculable upper bound, as expressed in (8):

$$T_{max} = N_{max} \cdot T_{timeout} + \sum_{n=1}^{N_{max}-1} B(n) \quad (8)$$

where T_{max} represents the theoretical upper bound of the total orchestration time within a single session. With $N_{max} = 3$, $T_{timeout} = 10$ s, and $B_{base} = 1.5$ s, we obtain $T_{max} = 34.5$ s. It is this strictly bounded characteristic that guarantees students never wait indefinitely, regardless of how degraded network conditions become. It should be noted that T_{max} is an upper bound rarely reached in practice, because as reported in Subsection III-A, failures at the testing site were dominated by instantaneous drops with a median of 29 ms, such that the empirical orchestration time of an actual three-attempt chain averaged only 5.71 seconds.

B. Reactive Object Detection Substitution Based on Orchestrator Outcome

Unlike proactive selection schemes that choose inference sources based on network status classification prior to dispatching a request, NARV implements a reactive substitution mechanism that depends entirely on the final outcome of the orchestrator in Subsection A. This design choice is substantiated by the jitter findings in Subsection III-A, which demonstrate that network conditions at the testing location fluctuate too severely between sessions to serve as a reliable predictive baseline.

The on-device object detection model runs continuously and independently of network conditions throughout the scanning phase, accumulating the set of detected object labels up to time t , as expressed in (9):

$$L_{yolo}(t) = L_{yolo}(t-1) \cup \text{YOLO}_{onDevice}(I_t) \quad (9)$$

where $L_{yolo}(t)$ denotes the set of unique labels accumulated up to time t , I_t represents the camera frame at time t , and $\text{YOLO}_{onDevice}(\cdot)$ is the local detection function,

implemented via Lightship in AR mode and via WebCamTexture paired with the YOLO model in non-AR mode. This process is purely local and neither awaits nor depends on network availability.

Upon trigger activation, specifically at time t^* when the instructional scanning interval concludes, a single frame capture I^* is acquired and dispatched to the cloud service through the orchestration process detailed in Subsection A. This transmission is executed unconditionally, regardless of whether $L_{yolo}(t^*)$ already contains labels or remains empty. This provision is vital, as the class vocabulary of the on-device detector is bounded, such that establishing $L_{yolo}(t^*) \neq \emptyset$ as a prerequisite would cause the session to stall whenever scanned objects fall outside that vocabulary. The final object detection output is defined as expressed in (10):

$$D = \begin{cases} L_{n^*}, & \text{if } \exists n^* = \min\{n \leq N_{max} \mid \sigma(n) = 1\} \\ L_{yolo}(t^*), & \text{if } \sigma(n) = 0 \quad \forall n \leq N_{max} \end{cases} \quad (10)$$

where D denotes the set of final labels presented to the student, n^* represents the index of the first successful attempt, and L_{n^*} is the set of labels generated during that attempt.

Under this formulation, the on-device detector does not act as an alternative inference pathway re-invoked upon failure, but rather as a concurrent background process whose results are pre-accumulated and primed for instantaneous substitution the moment all cloud attempts are declared failed, introducing zero additional latency at the precise moment of substitution. It must also be emphasized that substitution is exhaustive rather than an aggregation, because when the cloud pathway succeeds, its output entirely supersedes $L_{yolo}(t^*)$. This constraint prevents the contamination of limited-vocabulary labels into the higher-fidelity result.

From (10), the continuity guarantee of the NARV pipeline is derived, as expressed in (11):

$$D \neq \emptyset \iff (\exists n \leq N_{max} \mid \sigma(n) = 1) \vee (L_{yolo}(t^*) \neq \emptyset) \quad (11)$$

The formulation in (11) establishes that a session fails to produce output if and only if all cloud attempts fail and the on-device detector concurrently detects no objects whatsoever. These two conditions are mutually independent, as the former depends on network connectivity whereas the latter is governed by the visual contents within the camera field of view. Across the experimental trials reported in Subsection III-B, the co-occurrence of these two conditions never transpired, thereby yielding $R_{cont} = 100.0\%$. In the theoretical event that such a combination occurs, the system does not present an empty result, but instead returns the session to the scanning phase.

C. Algorithmic Execution Flow

The overarching coordination between the continuously running on-device detection process and the momentarily triggered cloud orchestrator is executed according to the procedure described in Algorithm 1, which comprises two concurrent processes.

Algorithm 1 Network-State Adaptive Resilient Vision (NARV) Flow

Input: Camera frame stream $\{I_t\}$; cloud endpoint E ; N_{max} ; $T_{timeout}$; B_{base}
Output: D final object detection result; $Source \in \{\text{Cloud}, \text{OnDevice}\}$

Process 1 (runs continuously throughout the scanning phase)

```

1:  $L_{yolo} \leftarrow \emptyset$ 
2: while scanning phase is active do
3:    $I_t \leftarrow \text{CURRENTCAMERAFRAME}()$ 
4:    $L_{yolo} \leftarrow L_{yolo} \cup \text{YOLO}_{onDevice}(I_t)$ 
5: end while

```

Process 2 (triggered once when the instructional interval elapses)

```

6:  $I^* \leftarrow \text{CAPTURESCREENFRAME}()$ 
7:  $P \leftarrow \text{BUILDMULTIMODALREQUEST}(I^*, L_{yolo})$ 
8:  $n \leftarrow 0$ ;  $success \leftarrow \text{false}$ 
9: while  $n < N_{max}$  and  $success = \text{false}$  do
10:    $n \leftarrow n + 1$ 
11:    $R_n \leftarrow \text{INVOKEVISIONAPI}(P, E, T_{timeout})$ 
12:   if  $R_n$  received within  $T_{timeout}$  then
13:      $L_n \leftarrow \text{PARSEOBJECTLIST}(R_n)$ 
14:     if  $L_n$  parsed successfully and  $|L_n| \geq 1$  then
15:        $D \leftarrow L_n$ ;  $success \leftarrow \text{true}$ 
16:        $\text{LOGEVENT}(\text{Vision\_Success})$ 
17:     else
18:        $\text{LOGEVENT}(\text{Vision\_Empty\_Response\_Error}, n)$ 
19:     end if
20:   else
21:      $\text{LOGEVENT}(\text{Vision\_API\_Error}, n)$ 
22:   end if
23:   if  $success = \text{false}$  and  $n < N_{max}$  then
24:      $\text{LOGEVENT}(\text{Vision\_Retry\_Scheduled}, B_{base} \cdot n)$ 
25:      $\text{WAIT}(B_{base} \cdot n)$ 
26:   end if
27: end while
28: if  $success$  then
29:   return  $D$ , "Cloud"
30: else
31:    $\text{LOGEVENT}(\text{Vision\_Fallback\_To\_YOLO})$ 
32:   if  $L_{yolo} \neq \emptyset$  then
33:     return  $L_{yolo}$ , "OnDevice"
34:   else
35:     revert to the Scanning phase
36:   end if
37: end if

```

The execution of Process 1 runs throughout the scanning phase entirely independent of network connectivity, continuously updating the label set L_{yolo} as specified in Lines 1 to 5. Once the instructional scanning interval concludes, Process 2 is triggered at Line 6 and captures a single screen frame, which is subsequently constructed into a multimodal request alongside L_{yolo} as initial context at Line 7. As long as n has not reached N_{max} and the attempt has not yet succeeded, the system continuously invokes the cloud service subject to a $T_{timeout}$ limit per attempt as defined in Lines 9 to 22, with the provision that both empty and unparseable responses are treated as failures according to (6). Any failure that still leaves subsequent attempts is handled through a linear backoff delay of $B_{base} \times n$ in Lines 23 to 26. If any attempt succeeds, the cloud results are returned as a cloud-sourced output in Lines 28 and 29. If all N_{max} attempts fail, the system immediately returns L_{yolo} , pre-accumulated from the onset, as a substitution without any additional delay in Lines 31 to 33, and only if this set is likewise empty is the session reverted to the scanning phase at Line 35.

V. IMPLICATIONS, FUTURE WORK, AND CONCLUSION

The empirical findings documented in this research establish a strong case for the deployment of reactive substitution architectures to ensure the continuity of AI-based object recognition on resource-constrained devices, particularly within underserved pedagogical environments.

This section contextualizes these technical outcomes within the broader socio-technical landscape and outlines the inherent limitations and future trajectories of this work.

A. Socio-Technical Implications for Underserved Regions

Deploying Mobile Augmented Reality and generative artificial intelligence in rural regions such as Sumbawa Besar presents a classic socio-technical paradox, as communities in greatest need of high-quality instructional tools are simultaneously the most hindered by telecommunication infrastructure limitations. The empirical field data in this research indicate that the nature of these bottlenecks diverges from conventional assumptions. Out of 49 failed cloud calls, 47 calls (95.9%) failed instantaneously with a median of 29 ms, largely comprising DNS resolution failures, whereas only 2 calls reached the timeout boundary. The primary constraint is therefore not a sluggish connection, but rather periodic absence of connectivity, rendering strategies that extend timeouts or defer transmissions until connection recovery of no practical benefit to students actively awaiting answers.

The NARV pipeline successfully resolves this paradox. All 90 sessions yielded object identification without a single session terminating with no output, with 76 sessions resolved via the cloud pathway and 14 sessions via on-device substitution. Importantly, the substitution burden was distributed highly unevenly across sessions, ranging from zero to 11 out of 30 sessions on different devices. Without this pathway, failures would have been concentrated among a subset of students, resulting in one segment of the classroom losing access while another remained unaffected. The continuity guarantee provided by NARV is therefore not merely a technical accomplishment, but also a vital pillar of instructional equity within the same classroom.

The strictly bounded nature of the waiting time further reinforces its deployment feasibility. Although orchestrator parameters imply a theoretical upper bound of $T_{max} = 34.5$ s, the empirical time cost of a three-attempt chain in the field amounted to only $M = 5.71$ s, because failed attempts returned within tens of milliseconds. Students are thereby never left waiting indefinitely, regardless of how degraded network conditions become.

B. Limitations and Future Work

Several technical limitations must be acknowledged to guide future iterations:

1. **Bounded Granularity on the Substitution Pathway:** The continuity achieved was not attained without trade-offs. The cloud pathway yielded 2.55 objects per session across a vocabulary of 49 distinct labels, whereas the substitution pathway produced only 2.00 objects per session limited to 10 broad categories, occasionally misclassifying objects outside the classroom context. The system guarantees that a session invariably produces an answer but does not guarantee that the answer possesses the semantic richness of the cloud output.
2. **Rendering Fluidity on Lower-Tier Devices:** The memory footprint remained lightweight and stable, with a per-session growth of $M = 11.86$ MB and a maximum proportional utilization of 7.43%. However, 46.8% of telemetry samples on the Entry-Mid device and 47.3%

on the Entry-Level device fell below the 18 FPS threshold, and this constraint propagated into the orchestrator through five backoff schedulings delayed by up to 1.335 seconds.

3. **Between-Subjects Design and Testing Confounders:** Each participant utilized only a single device, the three devices were evaluated across different time intervals, and the deployment of AR versus non-AR modes was governed by device capabilities rather than randomized assignment. Cross-device comparisons are therefore descriptive in nature and cannot be interpreted as causal relationships.
4. **Location-Specific Sensitivity of Pathway Proportions:** Out of the 18 sessions requiring retries, 4 sessions were resolved through retry and 14 sessions via substitution. This proportion reflects the persistent nature of failures at the testing site, dominated by DNS resolution drops, meaning that in environments with more transient failure patterns, their relative contributions could potentially differ.

Future research will explore deploying lightweight vision-language models directly on-device via ONNX Runtime to narrow the granularity gap, optimizing rendering budgets on lower-tier devices, and implementing a within-subjects design to isolate the effects of hardware tiers from temporal network fluctuations.

C. Conclusion

This paper presented the design, implementation, and empirical evaluation of the Network-State Adaptive Resilient Vision (NARV) pipeline within the RoboAR application, resolving AI-based object recognition reliability in network-constrained environments. By coupling linear backoff cloud orchestration with reactive on-device substitution, NARV guarantees continuous instructional output independent of connectivity availability. In-the-wild evaluations across 90 sessions from 90 participants on three hardware tiers achieved a 100.0% Continuity Rate, resolving 84.4% of sessions via the cloud and 15.6% via on-device substitution. Furthermore, backoff scheduling was empirically validated against the $B(n) = B_{base} \cdot n$ model, with 30 out of 35 schedulings executing on time within a mean deviation of 0.038 seconds. Regarding device feasibility, the pipeline maintained a stable memory footprint, exhibiting an average growth of $M = 11.86$ MB per session and a peak proportional utilization of 7.43% on the Entry-Level tier. Crucially, empirical failure logging revealed that 95.9% of cloud drops were instantaneous with a median of 29 ms, demonstrating that system architectures for underserved regions must prioritize preparedness for complete service absence over mere latency tolerance. Ultimately, this work proves that immersive AI educational tools can operate reliably and equitably in underserved regions, paving the way for resilient, locally grounded pedagogical companions on the mobile edge.

REFERENCES

- [1] C. Papakostas, C. Troussas, A. Krouska, and C. Sgouropoulou, "Exploring Users' Behavioral Intention to Adopt Mobile Augmented Reality in Education through an Extended Technology Acceptance Model," *Int. J. Hum. Comput. Interact.*, vol. 39, no. 6, pp. 1294–1302, 2023, doi: 10.1080/10447318.2022.2062551.

- [2] Z. Hu, "Mitigating the Urban-rural Digital Divide: A Dual Scaffolding-embedded Mobile Augmented Reality Learning Approach in the Post COVID-19 Pandemic," *Educational Technology & Society*, vol. 26, no. 4, pp. 108–122, 2023, doi: 10.2307/48747524.
- [3] G. Makransky and G. B. Petersen, "The Theory of Immersive Collaborative Learning (TICOL)," *Educ. Psychol. Rev.*, vol. 35, no. 4, 2023, doi: 10.1007/s10648-023-09822-5.
- [4] A. Norman, "Educational technology for reading instruction in developing countries: A systematic literature review," *Review of Education*, vol. 11, 2023, doi: 10.1002/rev3.3423.
- [5] V. Sisouvang and K. Pasanchay, "Mobile Learning: Enhancing Self-Directed Education through Technology, Wireless Networks, and the Internet Anytime, Anywhere," *Journal of Education and Learning Reviews*, vol. 1, no. 2, pp. 39–50, 2024, doi: 10.60027/jelr.2024.752.
- [6] J. Ferreira-Santos and L. Pombo, "The Art Nouveau Path: Requirements Engineering and Traceability for City-Scale In-the-Wild Mobile Augmented Reality Learning Services," *Computers*, vol. 15, no. 4, 2026, doi: 10.3390/computers15040243.
- [7] L. Huang et al., "A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions," *ACM Trans. Inf. Syst.*, vol. 43, no. 2, 2025, doi: 10.1145/3703155.
- [8] H. Liu et al., "A Survey on Hallucination in Large Vision-Language Models," 2024, [Online]. Available: <http://arxiv.org/abs/2402.00253>
- [9] S. Priyadharma, "'Periphery-Centric' Approach as a Tactic for Everyday Digital In- and Exclusion of Indonesian Villages," *Media Commun.*, vol. 12, 2024, doi: 10.17645/mac.8162.
- [10] A. C. Situmorang, M. Suryanegara, D. Gunawan, and F. H. Juwono, "Proposal of the Indonesian Framework for Telecommunications Infrastructure Based on Network and Socioeconomic Indicators," *Informatics*, vol. 10, no. 2, pp. 1–33, 2023, doi: 10.3390/informatics10020044.
- [11] S. N. Jafari and B. M. C. Silva, "A Survey on Real-Time Data Transfer and Energy Consumption Strategies for Rural and Remote IoT Technologies," *IEEE Open Journal of the Computer Society*, vol. 6, 2025, doi: 10.1109/OJCS.2025.3619210.
- [12] K. Jannes, E. H. Beni, B. Lagaisse, and W. Joosen, "BeauForT: Robust Byzantine Fault Tolerance for Client-Centric Mobile Web Applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 4, pp. 1241–1252, 2023, doi: 10.1109/TPDS.2023.3241963.
- [13] P. Kumar, P. Kumar Patnaik, G. Dixit, A. Kumar, and K. Business School, "Additional information Versions of research works A I S sssociation for nformation ystems Digital Divide among Marginalized Rural Communities in Developing Countries: Strategies and Practices to Reduce the 'Proxy Use of ICTs' for Rural E-governance Thanos Papadopoulos," vol. 44, pp. 123–179, 2025, doi: 10.17705/1CAIS.044XX.
- [14] M. Giannakos et al., "The promise and challenges of generative AI in education," *Behaviour and Information Technology*, vol. 44, no. 11, pp. 2518–2544, 2025, doi: 10.1080/0144929X.2024.2394886.
- [15] S. Chen, S. Zhao, and C. Huang, "An Automatic Malaria Disease Diagnosis Framework Integrating Blockchain-Enabled Cloud-Edge Computing and Deep Learning," *IEEE Internet Things J.*, vol. 10, no. 24, 2023, doi: 10.1109/IIOT.2023.3304526.
- [16] E. Olaye and D. Obuh, "An Offline-First Mobile Reporting System for Digital One Health Surveillance in Resource-Constrained Settings," *International Journal of Applied Methods in Electronics and Computers*, vol. 14, no. 2, pp. 81–88, 2026, doi: 10.58190/ijamec.2026.172.
- [17] H. I. Liu et al., "Lightweight Deep Learning for Resource-Constrained Environments: A Survey," *ACM Comput. Surv.*, vol. 56, no. 10, 2024, doi: 10.1145/3657282.
- [18] A. S. Tiwari, K. K. Bhagat, and G. Lampropoulos, "Designing and evaluating an augmented reality system for an engineering drawing course," *Smart Learning Environments*, vol. 11, no. 1, pp. 1–19, 2024, doi: 10.1186/s40561-023-00289-z.
- [19] A. Pollini and G. A. Giacobone, "Unsustainability in Sustainability Education: Limits of Technology In Situ," *Sustainability (Switzerland)*, vol. 17, no. 20, 2025, doi: 10.3390/su17209178.
- [20] Y. Fu, P. Qin, and C. Jiang, "Edge AI Inference in 6G Mobile Networks: From Communication-Efficient Techniques to Task-Oriented Optimization," *IEEE Communications Surveys and Tutorials*, vol. 28, pp. 6174–6210, 2026, doi: 10.1109/COMST.2026.3690731.
- [21] R. Kaviyaraj and U. Mohan, "An Augmented Reality Framework for Education: Deep Learning Integration and Impact Evaluation," *IEEE Access*, vol. 13, 2025, doi: 10.1109/ACCESS.2025.3551656.
- [22] W. Tarnq, J. K. Huang, and K. L. Ou, "Improving Elementary Students' Geometric Understanding Through Augmented Reality and Its Performance Evaluation," *Systems*, vol. 12, pp. 1–28, 2024, doi: 10.3390/systems12110493.
- [23] L. Lv and S. Yao, "A robust detect and describe framework for object recognition in early childhood education," *PeerJ Comput. Sci.*, vol. 11, 2025, doi: 10.7717/peerj-cs.3080.
- [24] A. Sharshar, L. U. Khan, W. Ullah, and M. Guizani, "Vision-Language Models for Edge Networks: A Comprehensive Survey," *IEEE Internet Things J.*, vol. 12, no. 16, pp. 32701–32724, 2025, doi: 10.1109/IIOT.2025.3579032.
- [25] S. Li et al., "Collaborative Inference and Learning between Edge SLMs and Cloud LLMs: A Survey of Algorithms, Execution, and Open Challenges," *ACM Comput. Surv.*, 2026, doi: 10.1145/3838593.
- [26] H. I. Ashqar, A. Jaber, T. I. Alhadidi, and M. Elhenawy, "Advancing Object Detection in Transportation with Multimodal Large Language Models (MLLMs): A Comprehensive Review and Empirical Testing," *Computation*, vol. 13, 2025, doi: 10.3390/computation13060133.
- [27] Y. Huang et al., "Vinci: A Real-time Smart Assistant Based on Egocentric Vision-language Model for Portable Devices," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 9, no. 3, 2025, doi: 10.1145/3749513.
- [28] S. Kalam, A. Kalakech, T. Shahzad, Z. Kobti, and M. Akhshik, "Insectopedia: An On-Device Mobile Pipeline for Fine-Grained Agricultural Pest Identification," in *Proceedings - 2026 13th International Conference on Future Internet of Things and Cloud, FiCloud 2026*, Institute of Electrical and Electronics Engineers Inc., 2026, pp. 463–468. doi: 10.1109/FiCloud70576.2026.00072.
- [29] Y. W. Syaifudin et al., "Dynamic Feature Modules for Scalable and Modular Mobile Applications: A Case Study in Smart Agriculture," in *2025 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies, 3ICT 2025*, Institute of Electrical and Electronics Engineers Inc., 2025. doi: 10.1109/3ict68299.2025.11442085.
- [30] Y. Hu, Z. Li, Y. R. Chen, Y. Cheng, Z. Cao, and J. Liu, "Content-Aware Adaptive Device-Cloud Collaborative Inference for Object Detection," *IEEE Internet Things J.*, vol. 10, no. 21, pp. 19087–19101, 2023, doi: 10.1109/IIOT.2023.3279579.
- [31] H. Zou et al., "Logical Correction Enabled Collaborative Person Detection Inference in Edge Networks," *IEEE Trans. Mob. Comput.*, vol. 25, no. 4, pp. 4747–4761, 2026, doi: 10.1109/TMC.2025.3626734.