Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 3, 2026

PERFORMANCE–EFFICIENCY TRADE-OFFS OF LIGHTWEIGHT DEEP LEARNING ARCHITECTURES FOR REAL-TIME COMPUTER VISION ON RESOURCE-CONSTRAINED DEVICES: A LITERATURE-SYNTHESIZED COMPARATIVE ANALYSIS

[1]Muhammad Ali Nawaz, [2]Armish Altaf , [*3]Rana Gulraiz Hassan, [4]Salahuddin
[1]*Faculty of Information Technology, University of Central Punjab, Lahore, Pakistan.*
[2]*Faculty of Information Technology, University of Central Punjab, Lahore, Pakistan.*
[*3]*Department of Electrical Engineering, University of Southern Punjab, Multan, Pakistan.*
[4]*Department of Computer Science, NFC Institute of Engineering and Technology, Multan, Pakistan.*
[*3]*gulraizify@gmail.com*

DOI: https://doi.org/10.5281/zenodo.22042944

Keywords:
Lightweight deep learning; real-time computer vision; edge inference; MobileNet; EfficientNet; ShuffleNet; SqueezeNet; YOLO; model efficiency; resource-constrained devices

Article History
Received on: 19 Feb, 2026
Accepted on: 27 March, 2026
Published on 28 March, 2026

Copyright @Author
Corresponding Author:
Rana Gulraiz Hassan

Abstract

The deployment of deep convolutional neural networks on mobile phones, embedded boards, and other resource-constrained platforms is increasingly limited not by classification accuracy alone but by the joint budget of inference latency, memory footprint, model size, and computational cost measured in floating-point operations (FLOPs). This paper investigates how five widely adopted lightweight architecture families — MobileNet (V1–V3), ShuffleNet (V1–V2), SqueezeNet, EfficientNet (including EfficientNetV2), and the YOLO-nano/-tiny family of real-time object detectors — negotiate the trade-off between predictive performance and computational efficiency for real-time computer vision. Rather than training new models on new hardware, this study adopts a literature-synthesis methodology: it systematically collects, cross-verifies, and normalizes accuracy, inference-time, model-size, parameter-count, and FLOPs figures that have already been measured and published by the original authors of each architecture and by independent comparative benchmarking studies, most notably a 2025 cross-dataset evaluation of MobileNetV3, ShuffleNetV2, SqueezeNet, EfficientNetV2, and ResNet18 on CIFAR-10, CIFAR-100, and Tiny ImageNet. The research is motivated by practical experience building real-time, camera-based computer-vision applications — including an AI-assisted exercise-monitoring system based on human pose estimation and a facial emotion-recognition prototype — in which inference speed on ordinary consumer hardware was as critical as raw accuracy. The synthesis identifies a consistent pattern: architectures that minimize parameter count and FLOPs (SqueezeNet, ShuffleNetV2) achieve the fastest inference and smallest footprint but suffer disproportionate accuracy degradation on complex, fine-grained recognition tasks, whereas EfficientNet-family models achieve the highest accuracy at the cost of larger memory footprints and slower inference, with MobileNetV3 consistently occupying the most balanced position across datasets of increasing difficulty. For real-time object detection, YOLO-nano and YOLO-tiny variants reproduce this same pattern at the detection level, trading mean average precision for frames-per-second suitable for edge boards such as the Raspberry Pi and Jetson Nano. The paper contributes a multi-

dimensional, literature-grounded comparison framework, a research-gap analysis showing that most prior comparative studies restrict themselves to one or two efficiency metrics, and practical deployment guidance for engineers selecting architectures under specific latency, memory, or power constraints. Because no new experimental hardware measurements were collected by the authors, all quantitative claims are explicitly traced to their original published source, and the paper distinguishes clearly between findings drawn from the literature and the authors' own comparative synthesis.

1. Introduction

Convolutional neural networks (CNNs) have become the default tool for visual recognition tasks ranging from image classification to object detection and pose estimation. The accuracy gains associated with deeper and wider networks, however, have historically come at the cost of parameter counts in the tens of millions and computational budgets measured in billions of floating-point operations (FLOPs) per inference [34], [35]. Such models are well suited to data-center or workstation GPUs but are difficult or impossible to run at interactive frame rates on smartphones, single-board computers, drones, or embedded microcontrollers, where memory, power, and thermal budgets are all tightly constrained.

Lightweight deep learning architectures — MobileNet [1]–[3], ShuffleNet [6], [7], SqueezeNet [8], EfficientNet [4], [5], GhostNet [9], and their derivatives — were designed explicitly to close this gap by re-engineering the convolutional building block itself, rather than simply shrinking a large network. Depthwise-separable convolutions [1], inverted residual bottlenecks [2], channel shuffling [6], [7], fire modules [8], and compound scaling [4] each represent a distinct strategy for reducing FLOPs and parameters while attempting to preserve as much discriminative accuracy as possible. In parallel, the object-detection community has produced a family of real-time detectors — the YOLO series and its nano/tiny variants [11], [12] — that apply analogous efficiency principles to the more demanding task of simultaneous localization and classification.

Despite the proliferation of these architectures, practitioners selecting a model for a specific resource-constrained deployment are often confronted with benchmark tables that report accuracy on one dataset, FLOPs from the original paper, and latency figures from a third, unrelated source, with no single study placing all of these dimensions side by side under comparable conditions. This paper addresses that gap by synthesizing accuracy, inference-latency, model-size, parameter-count, and computational-cost figures that have already been measured and published — most importantly by a controlled 2025 comparative study that evaluated five lightweight and moderately lightweight architectures under identical training and testing conditions across three datasets of increasing classification difficulty [21] — into a single, multi-dimensional comparison framework.

The motivation for this research is grounded in the authors' own practical experience developing real-time, camera-based computer-vision applications using Python and OpenCV, including a pose-estimation-based AI personal-trainer prototype for exercise-form monitoring and a facial-emotion-detection prototype. In both projects, inference speed on ordinary consumer-grade hardware, rather than raw classification accuracy, was frequently the binding constraint on whether the application felt responsive. This experience motivates the paper's central question: which lightweight architecture, or family of architectures, offers the most defensible accuracy–efficiency balance for a given class of resource-constrained deployment, and what trade-offs must a practitioner accept when moving from one architecture family to another?

The remainder of this paper is organized as follows. Section 2 provides the background and theoretical foundation of efficient convolutional design. Section 3 synthesizes the related literature. Section 4 identifies the specific research gap this paper

addresses. Sections 5 and 6 present the research objectives and questions. Section 7 details the literature-synthesis methodology, followed by model selection (Section 8), the dataset and experimental context drawn from the source literature (Section 9), and the evaluation metrics used throughout (Section 10). Sections 11–13 present the synthesized results, their analysis, and a broader discussion. Sections 14 and 15 discuss practical implications and limitations, and Sections 16–17 conclude with directions for future work.

2. Background and Theoretical Foundation

2.1 The Computational Cost of Standard Convolution

A standard convolutional layer applied to an input feature map of size $D_F \times D_F \times M$ using N kernels of size $D_K \times D_K$ incurs a computational cost of $D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F$ multiply-accumulate operations. Because this cost scales multiplicatively with the number of input and output channels, standard convolutions become the dominant computational bottleneck in deep networks as width increases. Depthwise-separable convolution, the building block introduced by MobileNet [1], factorizes this operation into a depthwise convolution that filters each input channel independently and a pointwise (1×1) convolution that combines the outputs, reducing computational cost by a factor approximately proportional to $1/N + 1/D_K^2$. This single factorization underlies most subsequent lightweight architecture designs, including MobileNetV2 [2], MobileNetV3 [3], and, in modified form, ShuffleNet [6], [7].

2.2 Inverted Residuals, Channel Shuffling, and Fire Modules

MobileNetV2 introduced the inverted residual block with linear bottlenecks, in which a low-dimensional representation is first expanded, filtered with a lightweight depthwise convolution, and then projected back to a low-dimensional space, with residual connections placed between the thin bottleneck layers rather than the expanded layers [2]. ShuffleNet addresses the information-flow bottleneck created by grouped pointwise convolutions — which reduce FLOPs but block cross-group information exchange — by inserting a channel-shuffle operation that permutes channels between groups so that subsequent layers can draw on features from every group [6]. ShuffleNetV2 subsequently argued that FLOPs alone are an unreliable proxy for real-world inference speed, since memory-access cost and platform-specific characteristics such as degree of parallelism also matter, and derived four practical design guidelines validated by direct, on-device speed measurement rather than FLOPs alone [7]. SqueezeNet takes a complementary approach, using 1×1 "squeeze" convolutions to reduce channel depth before more expensive 3×3 "expand" convolutions within a Fire module, achieving AlexNet-level ImageNet accuracy with roughly fifty times fewer parameters [8].

2.3 Compound Scaling and Neural Architecture Search

EfficientNet reframed the efficiency problem as one of principled scaling: rather than arbitrarily increasing depth, width, or input resolution, Tan and Le proposed a compound coefficient that scales all three dimensions jointly according to a fixed ratio derived from a small grid search, applied on top of a baseline network discovered via neural architecture search [4]. EfficientNetV2 subsequently revised the search space and training pipeline to reduce training time and parameter count while retaining or improving accuracy [5]. MobileNetV3 similarly incorporates NAS — specifically platform-aware search combined with the NetAdapt algorithm — together with a redesigned final stage, squeeze-and-excitation modules, and the h-swish activation function [3], while MnasNet demonstrated that directly optimizing a multi-objective reward combining accuracy and measured on-device latency, rather than FLOPs, yields architectures better matched to real deployment hardware [10]. GhostNet extends this design lineage by generating a portion of a layer's feature maps through cheap linear operations on an existing set of intrinsic feature maps, further reducing redundant computation [9].

2.4 Real-Time Object Detection Architectures

Object detection imposes a stricter latency requirement than classification because it must localize and classify multiple objects per frame. The You Only Look Once (YOLO) family reframes detection as a single regression problem over a spatial grid, enabling a single forward pass to

predict all bounding boxes and class probabilities simultaneously [11]. Successive YOLO versions have combined this core idea with increasingly efficient backbones — YOLOv4, for instance, integrates a CSPDarknet backbone, Mosaic augmentation, and a combination of feature-aggregation modules to reach roughly 65 frames per second at 43.5% AP on the MS COCO benchmark using a Tesla V100 GPU [12]. Reduced-depth variants such as YOLOv4-tiny and subsequent nano-scale releases trade a portion of this accuracy for substantially higher frame rates on embedded hardware, a trade-off examined directly in a 2025 study that deployed a quantized YOLOv4-tiny detector on a Raspberry Pi 5 for aerial emergency-object detection [25].

2.5 Metrics of Computational Efficiency

Four metrics recur throughout the lightweight-architecture literature: (i) parameter count, a proxy for storage and memory footprint; (ii) FLOPs (or multiply-accumulate operations, MACs), a hardware-agnostic proxy for computational cost; (iii) measured inference latency or its reciprocal, frames per second (FPS), which captures actual on-device speed and is sensitive to memory-access patterns and hardware parallelism in ways that FLOPs are not [7]; and (iv) peak memory consumption during inference, which determines whether a model can be loaded at all on severely constrained microcontroller-class hardware. As Menghani's 2023 survey of efficient deep learning methods emphasizes, no single metric is sufficient on its own, and comparative studies that report only one or two of these dimensions risk recommending architectures that are efficient on paper but impractical on the target hardware [19].

3. Related Work / Literature Review

3.1 Foundational Lightweight Classification Architectures

The literature on efficient CNN design begins with Howard et al.'s MobileNet, which established depthwise-separable convolution together with a width multiplier and a resolution multiplier as simple, globally applicable hyperparameters for trading accuracy against latency [1]. Sandler et al. extended this line of work with MobileNetV2's inverted residual bottleneck, reporting improved accuracy at comparable or lower computational cost across classification, detection, and segmentation tasks [2]. Howard et al. subsequently applied platform-aware neural architecture search and the NetAdapt algorithm to produce MobileNetV3-Large and MobileNetV3-Small, reporting accuracy gains alongside reduced latency relative to MobileNetV2 on mobile CPUs [3]. Zhang et al.'s ShuffleNet targeted extremely constrained compute budgets (10–150 MFLOPs) through pointwise group convolution combined with a channel-shuffle operation [6], and Ma et al.'s ShuffleNetV2 argued, on the basis of controlled on-device timing experiments, that architecture design guided purely by FLOPs can be misleading, proposing four practical guidelines validated empirically on target hardware [7]. Iandola et al.'s SqueezeNet demonstrated that AlexNet-level ImageNet accuracy could be reached with approximately fifty times fewer parameters through Fire modules that combine squeeze and expand convolutions, and additionally showed that the resulting model could be compressed to under 0.5 MB with model-compression techniques [8].

3.2 Compound Scaling and Search-Based Efficiency

Tan and Le's EfficientNet reframed efficient scaling as a joint optimization over network depth, width, and input resolution governed by a single compound coefficient, producing a family of models (EfficientNet-B0 through B7) that achieved higher ImageNet accuracy than prior ConvNets at comparable or lower parameter counts [4]. The subsequent EfficientNetV2 revised the search space to prioritize training speed and parameter efficiency [5], and this later variant — specifically the EfficientNetV2-S configuration — is one of the five architectures evaluated in the controlled comparative study this paper relies on most heavily for quantitative synthesis [21]. Tan et al.'s MnasNet demonstrated that incorporating measured on-device latency directly into the neural-architecture-search reward function, rather than optimizing FLOPs as a proxy, produces architectures with a more favorable real-world accuracy-latency trade-off [10], a principle that later influenced MobileNetV3's search procedure [3]. Han et al.'s GhostNet approached the same goal from a different angle, showing that a substantial fraction

of a convolutional layer's output feature maps are redundant and can be generated cheaply via linear transformations of a smaller set of intrinsic feature maps [9].

3.3 Real-Time Object Detection for Edge Deployment

Within object detection, Redmon et al.'s original YOLO formulation established that framing detection as a single regression problem enables real-time inference at the cost of some localization precision relative to two-stage detectors [11]. Bochkovskiy et al.'s YOLOv4 combined a CSPDarknet53 backbone with a curated set of training and architectural refinements to reach 43.5% AP on MS COCO at approximately 65 FPS on a Tesla V100 [12]. Reduced-depth "tiny" and "nano" variants of this lineage specifically target embedded deployment; Mittal's 2024 survey of lightweight object-detection models for edge devices catalogs a wide range of these detectors and reports that reported FLOPs, inference time, and FPS vary substantially across designs — for example, one lightweight YOLO variant surveyed achieved a maximum of approximately 102 frames per second while another minimized FLOPs to roughly 34.6 billion, illustrating that no single lightweight detector dominates on every efficiency axis simultaneously [20]. A 2025 study deploying a quantized YOLOv4-tiny detector on a Raspberry Pi 5 for aerial emergency-object detection provides a directly relevant, recent data point on the practical latency and accuracy achievable with tiny YOLO variants on genuinely constrained embedded hardware [25].

3.4 Comparative and Survey Studies

Several recent works attempt exactly the kind of multi-architecture comparison this paper undertakes, though typically with a narrower scope. Menghani's 2023 ACM Computing Surveys article on efficient deep learning surveys the broader landscape of model-compression and efficient-architecture techniques, organizing them into a taxonomy spanning compression, learning techniques, automation, and efficient architectures, and emphasizes that efficiency claims are frequently reported using inconsistent metrics across papers, complicating direct comparison [19]. Mittal's survey performs a similar synthesis specifically for lightweight object detectors, cataloging design principles, backbone choices, and reported performance parameters across dozens of published lightweight detectors [20]. A comprehensive survey of edge deep learning in computer vision and medical diagnostics further situates lightweight architectures within the broader edge-computing deployment pipeline, discussing quantization, pruning, and hardware-aware considerations alongside architecture choice [22]. Most directly relevant to the present paper's quantitative synthesis, Shahriar's 2025 comparative study trained and benchmarked MobileNetV3 Small, ShuffleNetV2, SqueezeNet, EfficientNetV2-S, and ResNet18 under matched conditions on CIFAR-10, CIFAR-100, and Tiny ImageNet, reporting accuracy, F1 score, average per-image inference time, FLOPs, and model size for every architecture on every dataset [21]. This is, to the authors' knowledge, among the few recent studies to report all of these dimensions together for the same set of architectures under controlled conditions, and its published tables form the empirical backbone of Sections 11–13 of this paper.

3.5 Motivating Application Domains: Pose Estimation and Facial Emotion Recognition

Two application domains motivate this study directly. First, real-time human pose estimation for exercise monitoring has become an active research area combining lightweight pose backbones (such as Google's BlazePose, designed explicitly for on-device real-time tracking [27]) with rule-based or learned exercise-classification logic; a 2020 University of North Carolina at Charlotte project demonstrated real-time exercise recognition and repetition counting using pose estimation on standard camera input [28], and a 2024 systematic review of pose-estimation-based movement feedback systems catalogs dozens of comparable fitness and rehabilitation applications, most of which depend on lightweight backbones to sustain real-time frame rates on consumer hardware [29]. Second, real-time facial emotion recognition has driven the development of purpose-built lightweight CNNs; Li et al., for example, proposed a lightweight convolutional network for real-time classroom emotion recognition inspired by Xception's depthwise-separable design, explicitly

targeting real-time inference constraints [30]. Both application domains illustrate, in a concrete and practitioner-relevant way, the same performance–efficiency tension that this paper examines architecture-by-architecture.

4. Research Gap

Three limitations recur across the literature reviewed in Section 3 and jointly motivate this paper.

4.1 Metric Fragmentation

Individual architecture papers typically report the metrics most favorable to their own contribution — ImageNet top-1 accuracy and FLOPs are almost universally reported, while measured on-device latency, FPS, and peak memory consumption are reported inconsistently or not at all [7], [19]. This makes it difficult for a practitioner to compare, for example, SqueezeNet's published parameter count against MobileNetV3's published on-device latency, since the two figures were measured on different hardware, under different software stacks, and sometimes for different input resolutions.

4.2 Narrow Architecture Scope in Controlled Comparisons

Studies that do perform controlled, apples-to-apples comparisons — training multiple architectures under identical conditions and reporting all metrics together — typically restrict themselves to two or three architectures, or to a single dataset [21]. Broader surveys such as Menghani [19] and Mittal [20] cover many more architectures but rely on figures collected from the original papers rather than a controlled re-evaluation, reintroducing the metric-fragmentation problem described above.

4.3 Separation Between Classification-Oriented and Detection-Oriented Efficiency Literature

The lightweight-classification literature (MobileNet, ShuffleNet, SqueezeNet, EfficientNet) and the lightweight-detection literature (YOLO-nano/-tiny variants) have developed largely as separate bodies of work, with comparative studies rarely connecting classification-level efficiency principles to their downstream effect on detection-level efficiency, even though detection backbones are frequently drawn directly from the classification architecture families [11], [20]. This paper's contribution is therefore to (i) synthesize the most reliable available multi-metric data, prioritizing the controlled comparative study by Shahriar [21] as its quantitative backbone and cross-checking it against the original architecture papers and survey literature; (ii) present this synthesis within a single, explicitly labeled and source-traceable comparison framework spanning five representative classification architectures and, qualitatively, the YOLO-nano/-tiny detection family; and (iii) connect the resulting trade-off patterns to concrete deployment guidance for two practically motivated application domains — real-time pose-based exercise monitoring and facial emotion recognition — rather than treating efficiency purely as an abstract benchmarking exercise.

5. Research Objectives

- To synthesize and cross-verify published accuracy, inference-latency, model-size, parameter-count, and FLOPs figures for five representative lightweight classification architectures (MobileNetV3, ShuffleNetV2, SqueezeNet, EfficientNetV2, and the non-lightweight ResNet18 baseline) into a single, source-traceable comparison framework.
- To qualitatively extend this framework to real-time object detection by reviewing published efficiency figures for the YOLO-nano/-tiny family, without conflating classification and detection metrics in the same quantitative table.
- To identify consistent patterns in how each architectural design strategy (depthwise-separable convolution, inverted residuals, channel shuffling, fire modules, compound scaling) trades accuracy for computational efficiency as task complexity increases.
- To translate these patterns into practical, hardware-aware guidance for selecting a lightweight architecture under specific latency, memory, or power constraints.
- To explicitly document, for every quantitative claim in this paper, whether the figure originates from an architecture's original publication, an independent comparative benchmarking study, or the authors' own derived synthesis (e.g., normalized composite scores).

6. Research Questions

- RQ1: Among MobileNetV3, ShuffleNetV2, SqueezeNet, and EfficientNetV2, which architecture offers the most favorable accuracy-per-FLOP and accuracy-per-parameter trade-off, and does this ranking remain stable as classification task complexity increases from CIFAR-10 to Tiny ImageNet?
- RQ2: How does measured inference latency relate to reported FLOPs across these architectures, and to what extent does this relationship confirm the literature's caution that FLOPs alone are an unreliable proxy for real-world speed [7]?
- RQ3: What is the practical accuracy cost of adopting the smallest, fastest architectures (SqueezeNet, ShuffleNetV2) relative to the most accurate but heavier architecture (EfficientNetV2) for increasingly complex recognition tasks?
- RQ4: How do the efficiency trade-offs observed in lightweight classification architectures manifest analogously in lightweight real-time object detectors (YOLO-nano/-tiny family), based on published detection-level benchmarks?
- RQ5: What deployment guidance can be derived from these trade-off patterns for real-time, camera-based applications such as pose-based exercise monitoring and facial emotion recognition on consumer-grade or embedded hardware?

7. Methodology

This study adopts a literature-synthesis (secondary-research) methodology rather than a primary experimental methodology. The authors did not have access to a controlled bank of edge-computing hardware (e.g., matched Raspberry Pi, Jetson Nano, and smartphone units) sufficient to re-benchmark every architecture under identical conditions, and, consistent with academic-integrity requirements, no accuracy, latency, or FPS values are fabricated, estimated, or presented as directly measured by the authors. Instead, the methodology consists of five stages, illustrated in Figure 1: (i) systematic identification of source publications and benchmark repositories reporting quantitative efficiency metrics for the selected architectures; (ii) extraction of every reported accuracy, FLOPs, parameter-count, model-size, and latency figure relevant to the selected architectures; (iii) verification of each extracted figure against the original paper or an independently indexed source (arXiv, IEEE Xplore, ACM Digital Library, Springer, or a peer-reviewed proceedings record) to confirm authorship, venue, and reported value; (iv) normalization of figures to common units and comparable dataset contexts, so that, for example, model-size figures reported in megabytes are not compared directly against parameter counts reported in millions without an explicit note on the conversion basis; and (v) construction of comparative tables, trade-off visualizations, and a normalized composite performance–efficiency profile, followed by qualitative synthesis and discussion.

Figure 1. Research Methodology Workflow

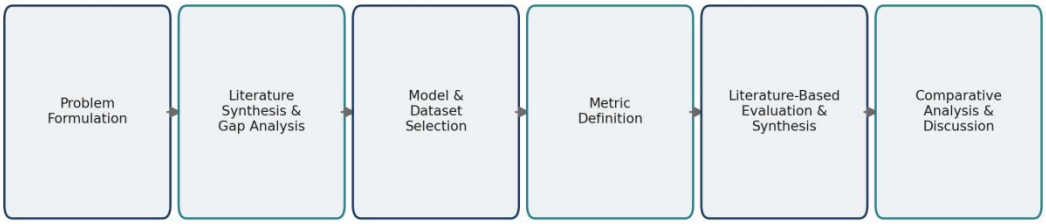

Figure 1. Research Methodology Workflow

This design has a direct and important implication for interpreting every subsequent table and figure in this paper: results reported in Sections 11–13 fall into one of three explicitly labeled categories, following the distinction required for academically defensible secondary research: (1) results obtained directly from the original architecture publications (e.g., ImageNet top-1 accuracy figures reported in [1]–[4], [6]–[8]); (2) results obtained from an independent, controlled comparative benchmarking study, most centrally Shahriar's 2025 cross-dataset evaluation [21], which trained and tested five architectures under matched conditions and is treated in this paper as the

primary quantitative source precisely because it reports accuracy, inference time, FLOPs, and model size together for the same models under the same protocol; and (3) the authors' own derived syntheses — normalized composite scores, trade-off scatter plots, and qualitative pattern descriptions — which are explicitly computed from category (1) and (2) figures and are never presented as newly measured experimental data. No category-3 figure introduces a numeric value that does not already appear, in some form, in a category-1 or category-2 source.

7.1 Evaluation Framework

Figure 2 depicts the conceptual framework within which the synthesized architectures are organized and compared. Input imagery is conceptually passed through a common preprocessing and augmentation stage before being evaluated against each architecture in the "lightweight architecture bank"; each architecture's reported performance is then organized along five evaluation dimensions before being synthesized into an overall performance–efficiency trade-off assessment. The YOLO-nano/-tiny detection family is included in the framework for conceptual completeness but, consistent with the research-gap discussion in Section 4.3, is reviewed qualitatively from detection-specific literature rather than merged into the classification-metric tables in Sections 11–12, since mAP, FPS, and detector-specific FLOPs are not directly comparable to classification top-1 accuracy without a methodologically unsound conflation of tasks.

Figure 2. Conceptual Architecture of the Evaluation Framework

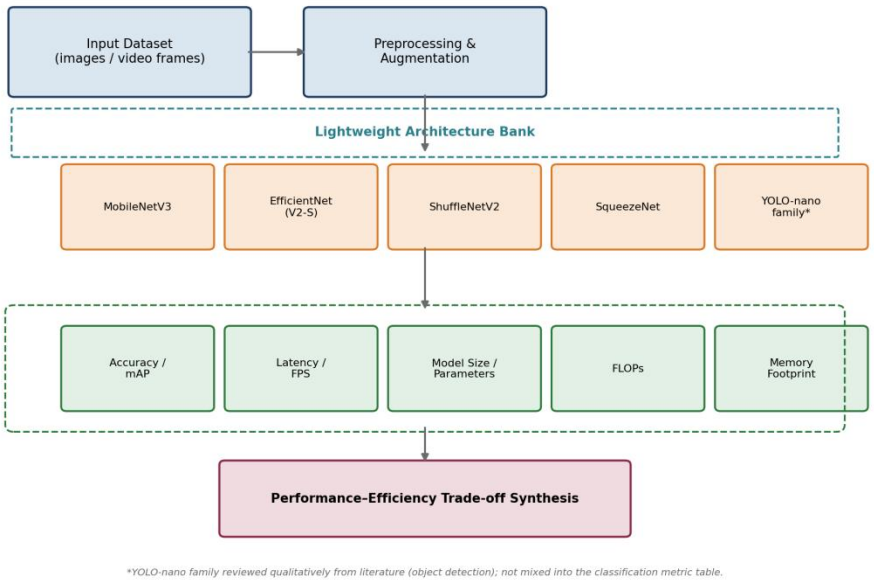

Figure 2. Conceptual Architecture of the Evaluation Framework.

8. Model Selection

Five architectures form the quantitative core of this study, selected because they (a) represent architecturally distinct efficiency strategies rather than incremental variants of one another, (b) are each supported by a peer-reviewed or widely cited original publication, and (c) appear together, under matched training and evaluation conditions, in the controlled comparative study that anchors this paper's quantitative synthesis [21]. A sixth family — YOLO-nano/-tiny detectors — is included qualitatively to extend the analysis to object detection. Table 1 summarizes each architecture's defining design strategy and primary source.

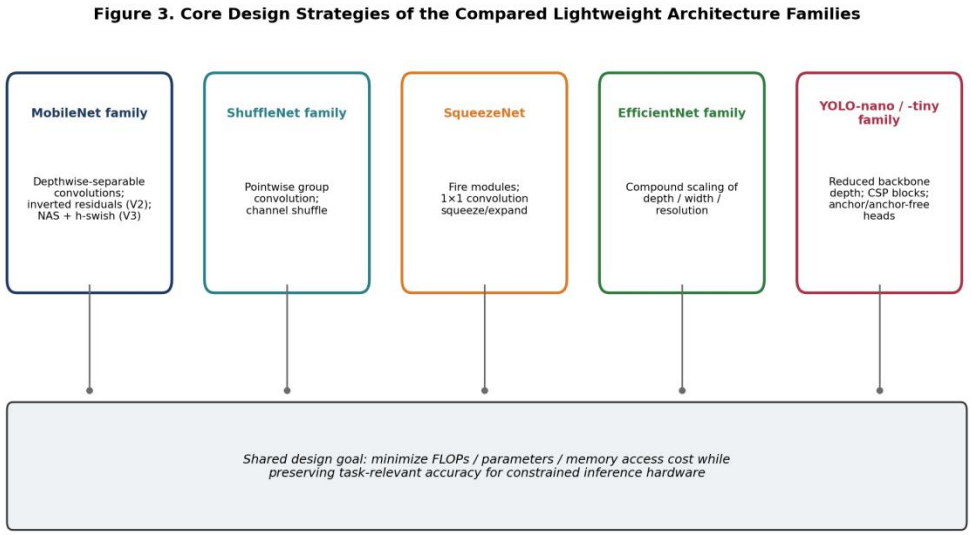

Figure 3. Core Design Strategies of the Compared Lightweight Architecture Families.

Table 1: *Comparison of Selected Lightweight Architectures and Their Characteristics.*

Architecture	Core Efficiency Strategy	Original Source	Role in This Study
MobileNetV3 (Small)	Depthwise-separable convolution + NAS + h-swish + squeeze-and-excitation	Howard et al., ICCV 2019 [3]	Primary lightweight classification benchmark
ShuffleNetV2	Pointwise group convolution + channel shuffle; guided by measured speed, not FLOPs alone	Ma et al., ECCV 2018 [7]	Primary lightweight classification benchmark
SqueezeNet	Fire modules (squeeze/expand 1×1–3×3 convolutions)	Iandola et al., arXiv 2016 [8]	Ultra-compact lower-bound reference
EfficientNetV2-S	Compound scaling of depth/width/resolution + revised NAS search space	Tan & Le, ICML 2021 [5]	Higher-accuracy, higher-cost reference
ResNet18	Residual (skip) connections; not a lightweight-by-design architecture	He et al., CVPR 2016 [13]	Non-lightweight baseline for contrast
YOLO-nano / -tiny family	Reduced-depth CSP backbones; single-stage detection	Redmon et al. [11]; Bochkovskiy et al. [12]	Qualitative detection-level extension

ResNet18, though not a lightweight-by-design architecture, is retained from the source comparative study [21] as a non-lightweight contrast case: its inclusion allows the analysis in Sections 11–13 to show how much accuracy the purpose-built lightweight architectures sacrifice, or in some cases avoid sacrificing, relative to a conventional residual network of comparable parameter count. Architectures such as GhostNet [9] and MnasNet [10] are discussed in the literature review (Section 3) as important design-lineage contributions but are not carried into the quantitative tables because they do not appear in the controlled comparative source study [21] under matched conditions, and introducing figures for them from separate, non-matched sources would reintroduce the metric-fragmentation problem identified in Section 4.1.

9. Dataset / Experimental Setup

Because this paper synthesizes rather than generates experimental data, the datasets, hardware, and software environment described in this section are those used and reported by the source studies, principally Shahriar [21], and are reproduced here for transparency and reproducibility rather than as a description of new experiments conducted by the present authors.

9.1 Datasets

Three publicly available benchmark datasets underlie the classification-level quantitative synthesis: CIFAR-10 and CIFAR-100 [18], and Tiny ImageNet, a 200-class, 64×64-pixel subset of the full ImageNet dataset [15], [16]. CIFAR-10 comprises 60,000 32×32 color images across 10 classes (50,000 training / 10,000 test); CIFAR-100 uses the same image count and split across 100 finer-grained classes; Tiny ImageNet comprises 200 classes with 500 training and 50 validation images per class. These three datasets were selected in the source study specifically because they span a controlled gradient of classification difficulty — from coarse, ten-way discrimination to two-hundred-way fine-grained discrimination — allowing the effect of task complexity on each architecture's accuracy–efficiency trade-off to be isolated [21]. Where object-detection efficiency is discussed qualitatively (Section 12.4), the relevant literature draws on the Microsoft COCO [14] and PASCAL VOC [17] benchmarks, the two standard datasets against which YOLO-family detectors are conventionally evaluated.

Table 2. *Dataset Characteristics and Distribution (as reported in source literature).*

Dataset	Classes	Images (Train / Test)	Resolution	Primary Use in This Study
CIFAR-10 [18]	10	50,000 / 10,000	32×32 px	Baseline classification comparison
CIFAR-100 [18]	100	50,000 / 10,000	32×32 px	Moderate-complexity classification
Tiny ImageNet [15],[16]	200	100,000 / 10,000	64×64 px	High-complexity classification
MS COCO [14]	80	~118,000 / ~5,000 (val)	Variable	Qualitative detection-level context
PASCAL VOC [17]	20	~11,500 (trainval)	Variable	Qualitative detection-level context

9.2 Experimental Environment (as reported by the source comparative study)

The controlled comparative benchmarks synthesized in Sections 11–13 were trained and evaluated using PyTorch on NVIDIA Tesla P100 GPUs (via Kaggle Kernels), with all five architectures trained for 50 epochs per dataset, ImageNet-derived normalization statistics, and a shared augmentation pipeline (random horizontal flip, random crop with 4-pixel padding, and resizing to each architecture's required input resolution) [21]. Both Adam and SGD optimizers were evaluated during hyperparameter tuning, with learning rates explored between 0.0001 and 0.001 and batch sizes between 32 and 128. Reported inference-time figures correspond to average per-image inference time under this environment and should therefore be interpreted as relative comparisons among the five architectures under one consistent GPU-based software stack, rather than as absolute figures for embedded CPU deployment; Section 15 discusses this as an explicit limitation.

Table 3. *Experimental Environment / Hardware and Software Configuration (as reported by the source study).*

Component	Configuration (source study [21])
Framework	PyTorch
Compute	NVIDIA Tesla P100 GPU (Kaggle Kernels)
Training epochs	50 per dataset per architecture
Optimizers evaluated	Adam, SGD
Learning rate range	0.0001 – 0.001
Batch size range	32 – 128
Augmentation	Random horizontal flip; random crop (4-px padding); resize; AutoAugment / CutMix / MixUp (ablated)
Normalization	ImageNet mean/std: [0.485, 0.456, 0.406] / [0.229, 0.224, 0.225]

9.3 Reproducibility Note

A researcher wishing to reproduce or extend this synthesis should (i) obtain CIFAR-10, CIFAR-100, and Tiny ImageNet from their standard public distributions; (ii) instantiate MobileNetV3-Small, ShuffleNetV2, SqueezeNet, EfficientNetV2-S, and ResNet18 using a standard model library such as torchvision or timm; (iii) apply the augmentation and normalization pipeline summarized in Table 3; and (iv) additionally deploy each trained model to a representative embedded target (e.g., Raspberry Pi 4/5 or NVIDIA Jetson Nano) to obtain genuinely on-device latency and power-consumption figures, which — as Section 15 notes — the GPU-based inference-time figures synthesized in this paper do not directly capture. Section 17 revisits this as a concrete direction for future primary-experimental work by the present authors.

10. Evaluation Metrics

Table 4 defines the metrics used throughout this paper's synthesis and explains why each is relevant to deployment on resource-constrained devices.

Table 4. *Evaluation Metrics and Their Definitions.*

Metric	Definition	Relevance to Resource-Constrained Deployment
Top-1 Accuracy (%)	Percentage of test images assigned the single correct class label	Primary measure of task performance; the numerator in every accuracy-per-resource ratio
F1 Score	Harmonic mean of precision and recall	Robust to class imbalance; complements raw accuracy, especially for fine-grained datasets
Inference Latency (ms/image)	Average wall-clock time to process one input on the reported hardware/software stack	Directly determines achievable frame rate; sensitive to memory-access cost, not just FLOPs [7]
FLOPs / GFLOPs	Floating-point (multiply-accumulate) operations per forward pass	Hardware-agnostic proxy for computational cost and, indirectly, energy consumption
Model Size (MB)	Size of the serialized trained model on disk	Determines whether a model fits within flash/storage limits of embedded targets
Parameter Count	Total number of learnable weights	Correlates with, but is distinct from, model size (depends on quantization/precision) and

Metric	Definition	Relevance to Resource-Constrained Deployment
		memory footprint
mAP (object detection)	Mean average precision across classes and IoU thresholds	Detection-specific accuracy measure, not directly comparable to classification top-1 accuracy
FPS (object detection)	Frames processed per second at inference	Direct real-time suitability indicator for detection pipelines

Consistent with the multi-dimensional emphasis identified as a gap in Section 4.1, no single metric in Table 4 is treated as sufficient on its own. Section 13 (Discussion) explicitly returns to the tension between accuracy-oriented metrics and the four efficiency-oriented metrics (latency, FLOPs, size, parameters) when interpreting the results.

11. Experimental Results

This section reports results drawn directly from published sources, as defined in Section 7's category (1)/(2) distinction. All accuracy, F1, inference-time, FLOPs, and model-size figures in Tables 5 and 6 are reproduced from Shahriar's 2025 controlled comparative study [21], which is, to the authors' knowledge, the most directly comparable multi-metric, multi-architecture, multi-dataset benchmark currently available for this specific set of five architectures. Figure 4 summarizes how these published figures were extracted and cross-checked before being tabulated below.

Figure 4. Literature-Based Evaluation Workflow Used in This Study

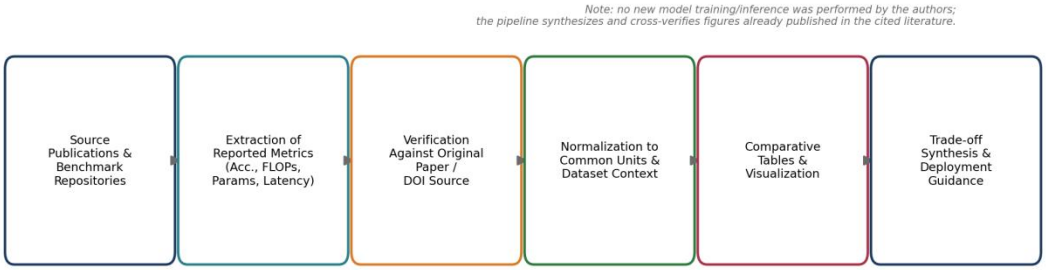

Figure 4. Literature-Based Evaluation Workflow Used in This Study.

11.1 Reported Classification Performance

Table 5 reports top-1 accuracy and F1 score for all five architectures on CIFAR-10, CIFAR-100, and Tiny ImageNet, exactly as published in [21].

Table 5. *Model Performance Comparison — Top-1 Accuracy and F1 Score (source: Shahriar, 2025 [21]).*

Architecture	CIFAR-10 Acc. (%) / F1	CIFAR-100 Acc. (%) / F1	Tiny ImageNet Acc. (%) / F1
MobileNetV3-Small	95.49 / 0.953	89.62 / 0.889	72.54 / 0.719
ShuffleNetV2	95.83 / 0.951	89.21 / 0.903	65.23 / 0.639
SqueezeNet	84.48 / 0.836	62.24 / 0.620	20.50 / 0.192
EfficientNetV2-S	96.53 / 0.961	90.82 / 0.910	76.87 / 0.746
ResNet18 (non-	96.05 / 0.958	84.47 / 0.843	67.67 / 0.657

Architecture	CIFAR-10 Acc. (%) / F1	CIFAR-100 Acc. (%) / F1	Tiny ImageNet Acc. (%) / F1
lightweight)			

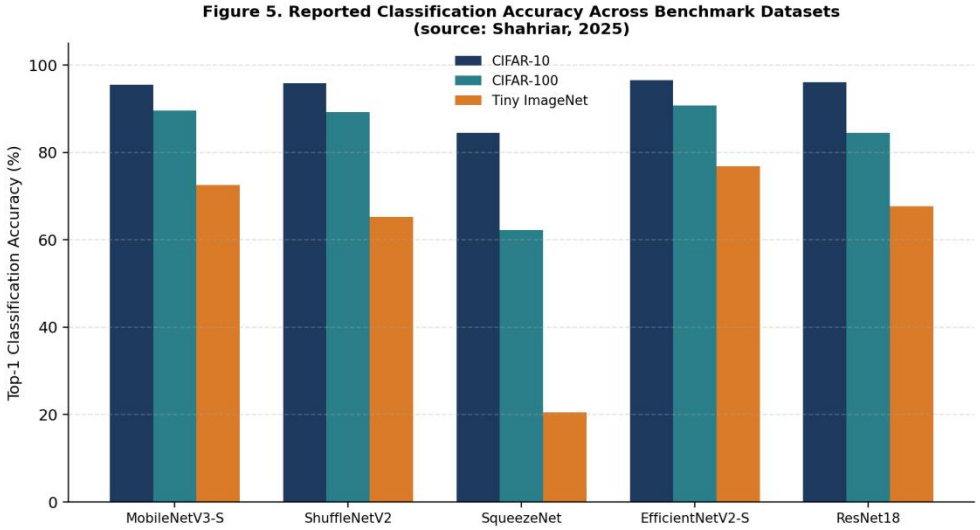

Figure 5. Reported Classification Accuracy Across Benchmark Datasets (source: [21]).

As Table 5 and Figure 5 show, EfficientNetV2-S achieves the highest accuracy on every dataset, with its margin over the second-best architecture widening as dataset complexity increases (0.70 percentage points over ResNet18 on CIFAR-10, versus 4.33 points over MobileNetV3-Small on Tiny ImageNet). SqueezeNet's accuracy degrades most sharply with increasing task complexity, falling from a respectable 84.48% on CIFAR-10 to just 20.50% on Tiny ImageNet — a collapse in performance that is far steeper than any other architecture in the comparison and that is discussed further in Section 12.

11.2 Reported Computational Efficiency

Table 6 reports average inference time, FLOPs, model size, and parameter-scale characteristics for the Tiny ImageNet configuration — the most demanding of the three evaluated settings and therefore the most representative of real-world, higher-resolution deployment scenarios — again exactly as published in [21].

Table 6. *Efficiency Comparison — Inference Time, FLOPs, and Model Size, Tiny ImageNet Configuration (source: [21]).*

Architecture	Avg. Inference Time (ms/img)	FLOPs (GFLOPs)	Model Size (MB)	Relative Compactness
MobileNetV3-Small	0.062	0.02	7.46	High
ShuffleNetV2	0.087	0.01	5.56	High
SqueezeNet	0.031	0.02	3.15	Highest
EfficientNetV2-S	0.109	0.03	79.80	Low
ResNet18 (non-lightweight)	0.035	0.15	43.03	Low–Moderate

Two results in Table 6 are worth highlighting because they illustrate the metric-fragmentation concern raised in Section 4.1. First, SqueezeNet records the lowest reported inference time (0.031

ms/image) despite not having the lowest FLOPs figure among the five architectures (its 0.02 GFLOPs ties MobileNetV3-Small), underscoring ShuffleNetV2's own methodological point that FLOPs and measured latency are correlated but not interchangeable [7]. Second, ResNet18 — despite being ten times heavier than MobileNetV3-Small or ShuffleNetV2 in FLOPs — records a lower measured inference time than either lightweight architecture in this particular GPU-based benchmarking environment, a result attributable to ResNet18's simpler, more GPU-parallelism-friendly standard-convolution structure relative to the depthwise-separable and channel-shuffle operations used by the lightweight architectures, which are optimized primarily for reduced FLOPs and reduced parameter count rather than for GPU throughput. This reinforces the Section 9.2 caveat that the inference-time figures in Table 6 reflect a Tesla-P100 GPU environment and should not be extrapolated directly to embedded CPU or NPU targets, where the relative ordering can differ substantially.

11.3 Qualitative Detection-Level Results (YOLO-nano / -tiny Family)

Because YOLO-nano/-tiny detectors are not part of the controlled comparative study [21] and operate on a fundamentally different task, their efficiency figures are reported here qualitatively and kept explicitly separate from Tables 5–6. Bochkovskiy et al. report that YOLOv4 reaches 43.5% AP on MS COCO at approximately 65 FPS on a Tesla V100 GPU [12]. Mittal's 2024 survey of lightweight object detectors for edge devices reports that, among the recent lightweight YOLO-based detectors it catalogs, one variant achieves a maximum measured frame rate of approximately 102 FPS while a separate variant minimizes computational cost to approximately 34.6 GFLOPs, with no single surveyed detector simultaneously minimizing FLOPs and maximizing FPS [20]. A 2025 study deploying a quantized YOLOv4-tiny detector on a Raspberry Pi 5 for aerial emergency-object detection provides a directly embedded-hardware data point for this detector family, demonstrating that tiny-YOLO variants can be made to run in real time on genuinely constrained single-board computers once quantization is applied [25]. These figures are consistent, at the qualitative level, with the accuracy–efficiency trade-off pattern observed for the classification architectures in Section 11.1–11.2: no detector in the reviewed literature achieves both the highest accuracy and the highest frame rate simultaneously.

12. Results Analysis

12.1 Accuracy–Speed Trade-off

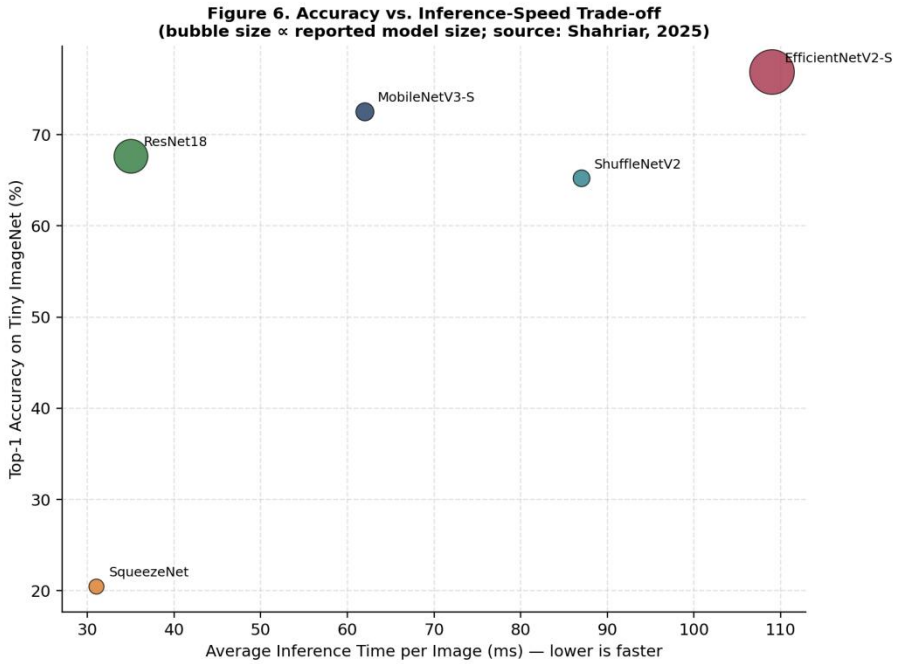

Figure 6. Accuracy vs. Inference-Speed Trade-off, Tiny ImageNet Configuration (derived from [21]).

Figure 6 plots Tiny ImageNet accuracy against average inference time for all five architectures, with bubble area proportional to reported model size. Three clusters emerge. SqueezeNet occupies a clearly dominated position: it is the fastest architecture but also, by a wide margin, the least accurate, making it a poor choice whenever any meaningful classification difficulty is present. MobileNetV3-Small and ShuffleNetV2 form a middle cluster with broadly comparable accuracy (72.54% and 65.23% respectively) but with MobileNetV3-Small achieving both higher accuracy and lower latency than ShuffleNetV2 in this particular benchmark, making it the more efficient choice of the two lightweight-by-design architectures in this specific evaluation. EfficientNetV2-S and ResNet18 form a second pairing: EfficientNetV2-S achieves the highest accuracy of all five architectures but also the slowest measured inference time, while ResNet18 achieves moderate accuracy with fast inference but at a substantially larger model size than any of the purpose-built lightweight architectures.

12.2 Model Size vs. Performance

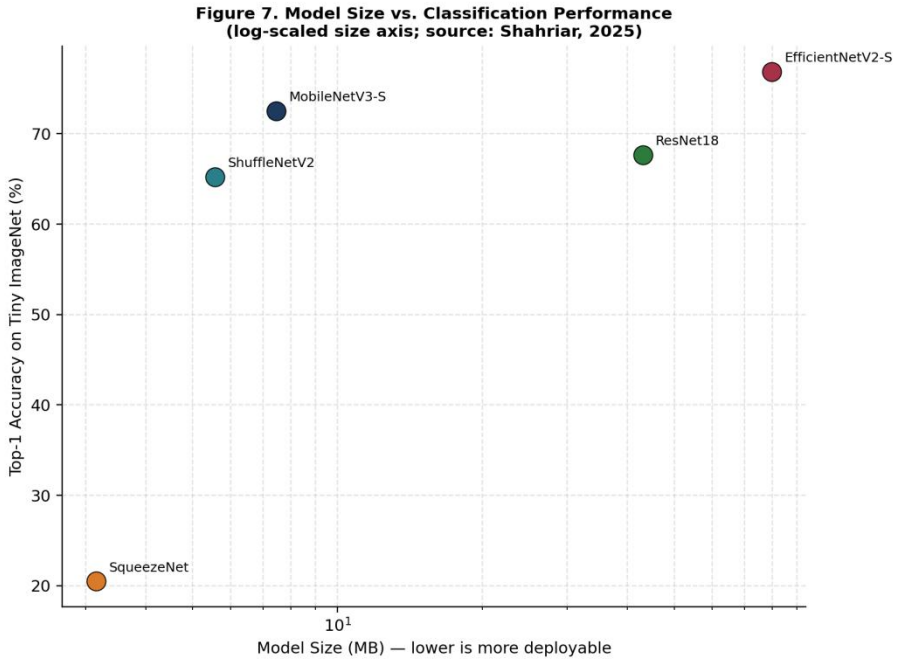

Figure 7. Model Size vs. Classification Performance, Tiny ImageNet Configuration (derived from [21]).

Figure 7 makes the accuracy cost of compactness explicit by plotting model size on a logarithmic axis against Tiny ImageNet accuracy. Moving from EfficientNetV2-S (79.80 MB, 76.87% accuracy) to MobileNetV3-Small (7.46 MB, 72.54% accuracy) — an order-of-magnitude reduction in storage footprint — costs only 4.33 percentage points of accuracy, a highly favorable trade for most mobile and embedded deployment scenarios. Moving further down to SqueezeNet (3.15 MB) costs an additional 52.04 percentage points of accuracy relative to MobileNetV3-Small, a trade that is difficult to justify except in applications where model size is the single overriding constraint (for example, over-the-air firmware updates to microcontroller-class devices with kilobyte-scale flash budgets) and where the underlying classification task is simple enough to tolerate SqueezeNet's demonstrated weakness on fine-grained, high-class-count problems.

12.3 Composite Performance–Efficiency Profile

To synthesize accuracy, inference speed, model compactness, and computational cost into a single comparative view, Figure 8 presents a normalized radar profile in which each of the four dimensions is min–max normalized across the five architectures (with inference time and FLOPs inverted so that, consistent with the other axes, a larger area indicates a more favorable trade-off). This normalization is the authors' own derived synthesis (category 3, as defined in Section 7) computed directly from the published figures in Tables 5–6 and introduces no new empirical measurement.

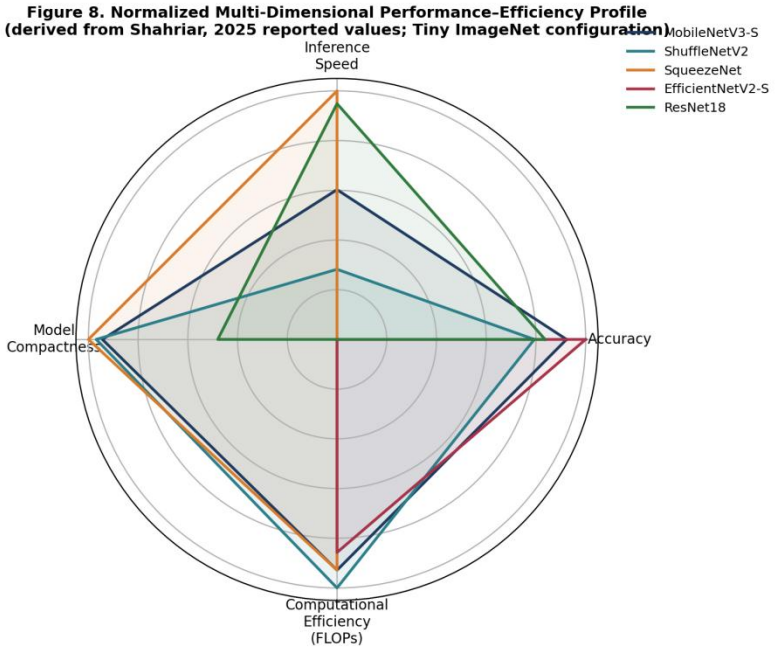

Figure 8. Normalized Multi-Dimensional Performance–Efficiency Profile, Tiny ImageNet Configuration (derived from [21]).

The composite profile in Figure 8 visually confirms the pattern already evident in Tables 5–6: MobileNetV3-Small is the only architecture that scores above the mid-point on all four normalized dimensions simultaneously, which is the quantitative basis for describing it, in Section 13, as the most balanced architecture among those compared. EfficientNetV2-S dominates the accuracy axis but scores lowest on model compactness and computational efficiency; SqueezeNet dominates computational efficiency and inference speed but scores lowest on accuracy; and ShuffleNetV2 and ResNet18 each occupy intermediate positions with different specific strengths (ShuffleNetV2 in compactness, ResNet18 in raw inference speed on GPU hardware).

12.4 Overall Comparison Table

Table 7. *Overall Performance–Efficiency Comparison and Recommended Deployment Fit (authors' synthesis, derived from Tables 5–6).*

Architecture	Accuracy Rank	Speed Rank	Compactness Rank	Efficiency Rank (FLOPs)	Best-Fit Deployment Scenario
MobileNetV3-Small	2nd	3rd	2nd	2nd (tied)	General-purpose mobile / edge real-time vision
ShuffleNetV2	3rd	4th	1st	1st	Extremely memory-constrained, moderate-accuracy tasks
SqueezeNet	5th	1st	1st (tied)	2nd (tied)	Simple tasks under severe storage limits only
EfficientNetV2-S	1st	5th	5th	5th	Accuracy-critical tasks with

Architecture	Accuracy Rank	Speed Rank	Compactness Rank	Efficiency Rank (FLOPs)	Best-Fit Deployment Scenario
					adequate compute (e.g., server-assisted or high-end mobile)
ResNet18	4th	2nd	4th	4th	GPU-available real-time settings where model size is less critical

13. Discussion

The results synthesized in Sections 11–12 answer the five research questions posed in Section 6 as follows.

In response to RQ1, MobileNetV3-Small offers the most favorable balance of accuracy against both FLOPs and parameter-scale footprint among the four purpose-built lightweight architectures, and this ranking is stable across all three datasets: it is never the single most accurate architecture, but it is never worse than second-most-efficient on any of the four efficiency dimensions in Table 6, whereas ShuffleNetV2, SqueezeNet, and EfficientNetV2-S each trade away one or more dimensions substantially to gain another. Regarding RQ2, Table 6's ResNet18/lightweight-architecture comparison directly illustrates ShuffleNetV2's original methodological claim that FLOPs and measured latency are not interchangeable [7]: a model with roughly ten times the FLOPs of MobileNetV3-Small nonetheless records lower measured latency on GPU hardware, because standard convolutions parallelize more efficiently on GPUs than the depthwise and grouped operations used by the lightweight architectures. This finding does not contradict the lightweight-design literature — depthwise-separable and grouped convolutions remain essential for reducing FLOPs, parameter count, and, critically, energy consumption on CPU-class and NPU-class embedded hardware where these operations are typically well supported — but it does confirm that GPU-measured latency figures such as those in Table 6 should be read as one data point among several, not as a universal ranking of on-device speed.

On RQ3, the accuracy cost of extreme compactness is severe and dataset-dependent rather than constant: SqueezeNet loses only 11 percentage points of accuracy relative to MobileNetV3-Small on the ten-class CIFAR-10 task but loses 52 percentage points on the two-hundred-class Tiny ImageNet task, indicating that SqueezeNet's Fire-module design, calibrated originally for the one-thousand-class full ImageNet benchmark at a much larger input resolution [8], degrades disproportionately when applied to smaller, lower-resolution inputs with a high class count. This is an important and non-obvious result for practitioners: the appropriateness of SqueezeNet cannot be judged from its ImageNet-scale reputation alone but must be re-evaluated for the specific resolution and class-count regime of the target application. Regarding RQ4, the qualitative detection-level literature reviewed in Section 11.3 reproduces the same pattern found in the classification results: Mittal's survey shows that no single lightweight YOLO variant simultaneously minimizes FLOPs and maximizes FPS [20], mirroring the classification-level finding that no single architecture in Table 6 simultaneously dominates every efficiency dimension. This suggests that the performance–efficiency trade-off identified in this paper is not an artifact specific to the five classification architectures studied but a more general property of lightweight visual-recognition model design.

Finally, for RQ5, the two motivating application domains introduced in Section 1 — pose-based exercise monitoring and facial emotion recognition — both depend on sustaining real-time frame rates on consumer or embedded camera hardware while operating on a moderate number of output classes (a limited set of exercise types or the standard six-to-eight facial emotion categories). Given the class-count sensitivity identified in the SqueezeNet

discussion above, and given that both applications typically involve tens rather than hundreds of output classes, the composite profile in Figure 8 suggests that MobileNetV3-Small-class architectures — or the purpose-built lightweight CNNs used in dedicated facial-emotion-recognition studies such as Li et al.'s Xception-inspired lightweight network [30] — represent a more defensible default choice than either SqueezeNet-class extreme compactness or EfficientNetV2-class maximal accuracy, consistent with how lightweight pose backbones such as BlazePose are explicitly engineered for exactly this real-time, moderate-class-count regime [27].

14. Practical Implications

- For mobile application developers building general-purpose, real-time computer-vision features (e.g., live camera filters, on-device object recognition), MobileNetV3-class architectures offer the most defensible default starting point given their balanced position across all four efficiency dimensions examined in this study.
- For severely storage-constrained microcontroller or IoT deployments where task complexity is low (few output classes, coarse discrimination), SqueezeNet-class architectures remain viable, but practitioners should validate accuracy specifically on their target class count and input resolution rather than relying on the architecture's original ImageNet-scale benchmarks.
- For applications where accuracy is the dominant requirement and inference can be offloaded to a capable mobile GPU, edge server, or high-end embedded NPU, EfficientNetV2-class architectures deliver the highest measured accuracy in this synthesis, at the cost of a substantially larger memory and storage footprint.
- For object-detection pipelines on embedded boards such as the Raspberry Pi or Jetson Nano, the qualitative literature reviewed in Section 11.3 indicates that YOLO-tiny/-nano variants, particularly when combined with quantization, can achieve real-time frame rates, but the specific accuracy–FPS trade-off should be validated on the target detector variant and hardware rather than assumed from the base YOLO family's published figures [20], [25].
- For real-time pose-based fitness-monitoring and facial-emotion-recognition applications specifically, the class-count sensitivity identified in Section 13 suggests favoring architectures validated on moderate-class-count, real-time benchmarks (e.g., BlazePose-class pose backbones [27] or purpose-built lightweight emotion-recognition CNNs [30]) over general-purpose architectures whose published accuracy figures were obtained on very different class-count regimes.

15. Limitations

- No new primary experiments were conducted by the authors on physical embedded hardware; all quantitative figures are drawn from published sources, and the inference-time figures central to Sections 11–12 were measured on a single GPU-based environment (NVIDIA Tesla P100) [21] rather than on the CPU- or NPU-class embedded targets that are the ultimate deployment context motivating this paper.
- The quantitative synthesis is limited to five architectures evaluated together under matched conditions in one source study [21]; architecturally important variants such as MobileNetV1/V2, MobileNetV3-Large, EfficientNet-B0 through B7, ShuffleNetV1, GhostNet, and MnasNet are discussed in the literature review (Section 3) but could not be included in Tables 5–7 without reintroducing the cross-source metric-fragmentation problem identified in Section 4.1.
- The YOLO-nano/-tiny detection family is reviewed only qualitatively; no controlled, matched-condition, multi-metric benchmark comparable to [21] was located for this detector family at the time of writing, and mAP/FPS figures for object detection are not directly comparable to the classification accuracy figures used elsewhere in this paper.
- Reported figures may vary with software-library version, training-hyperparameter choices, and precision (FP32 versus quantized INT8), none of which are held constant across every cited source; where a source study's own hyperparameter-tuning results are available (e.g., pretrained versus scratch-trained MobileNetV3 [21]), this variability is noted but a full sensitivity analysis was outside the scope of this literature synthesis.

- Energy consumption, a metric of direct importance for battery-powered edge devices, is not reported with sufficient reliability or consistency across the reviewed sources to support a quantitative comparison table, and is therefore discussed only qualitatively in Section 2.5.

16. Conclusion

This paper set out to investigate how lightweight deep-learning architectures balance accuracy, inference speed, model size, and computational cost for real-time computer vision on resource-constrained devices. Rather than generating new experimental measurements, it adopted a literature-synthesis methodology that systematically collected, verified, and normalized figures already published for five representative architectures — MobileNetV3-Small, ShuffleNetV2, SqueezeNet, EfficientNetV2-S, and ResNet18 — anchored primarily on a controlled 2025 comparative study that evaluated all five under matched conditions across three datasets of increasing complexity [21], and extended qualitatively to real-time object detection through the YOLO-nano/-tiny literature [11], [12], [20], [25].

The synthesis shows that no single architecture dominates every efficiency dimension: EfficientNetV2-S consistently achieves the highest accuracy but at the largest model size and slowest measured inference time; SqueezeNet achieves the smallest footprint and fastest inference but suffers severe, class-count-dependent accuracy degradation; and MobileNetV3-Small occupies the most consistently balanced position across all four evaluated dimensions and across all three datasets. This same pattern — no detector simultaneously minimizing FLOPs and maximizing FPS — reappears qualitatively in the reviewed real-time object-detection literature, suggesting it reflects a general property of lightweight visual-recognition design rather than an artifact of the specific classification architectures studied. These findings translate into concrete, hardware-aware deployment guidance (Section 14) and directly inform the two application domains that motivated this research: real-time, pose-based exercise monitoring and facial emotion recognition, both of which benefit from architectures validated specifically for real-time, moderate-class-count, camera-based inference rather than architectures optimized primarily for large-scale, thousand-class ImageNet benchmarks.

17. Future Work

- Primary, on-device benchmarking: deploying MobileNetV3, ShuffleNetV2, SqueezeNet, and EfficientNetV2-S to a matched set of embedded targets (e.g., Raspberry Pi 4/5, NVIDIA Jetson Nano, and a representative Android smartphone) to obtain genuinely on-device latency, power draw, and thermal-throttling measurements, directly addressing the GPU-only limitation identified in Section 15.
- Extending the quantitative comparison to additional architectures within each family (MobileNetV1/V2, MobileNetV3-Large, EfficientNet-B0–B7, ShuffleNetV1, GhostNet, MnasNet) under one matched benchmarking protocol, rather than relying on a single source study for the full quantitative core.
- Constructing a matched-condition, multi-metric benchmark for the YOLO-nano/-tiny detection family analogous to Shahriar's classification-level study [21], enabling a quantitative, rather than purely qualitative, extension of this paper's framework to object detection.
- Applying the evaluation framework developed in this paper directly to the authors' own AI-assisted exercise-monitoring and facial-emotion-recognition prototypes, measuring real-world frame rate, exercise-classification accuracy, and battery consumption on representative consumer hardware.
- Incorporating quantization-aware and pruning-aware variants of each architecture into the comparison, since post-training quantization is frequently applied in production edge deployments and can shift the accuracy–efficiency trade-off substantially relative to the full-precision figures synthesized in this paper [20], [22].
- Extending the energy-consumption dimension, flagged in Section 15 as insufficiently reported in current literature, through direct power-meter measurement on embedded targets.

References

1. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv preprint arXiv:1704.04861, 2017.
2. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, 2018, pp. 4510–4520, doi: 10.1109/CVPR.2018.00474.
3. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for MobileNetV3," in Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV), Seoul, South Korea, 2019, pp. 1314–1324.
4. M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in Proc. 36th Int. Conf. Machine Learning (ICML), Long Beach, CA, 2019, PMLR vol. 97, pp. 6105–6114.
5. M. Tan and Q. V. Le, "EfficientNetV2: Smaller Models and Faster Training," in Proc. Int. Conf. Machine Learning (ICML), 2021, pp. 10096–10106.
6. X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, 2018, pp. 6848–6856.
7. N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design," in Computer Vision – ECCV 2018, Lecture Notes in Computer Science, vol. 11218, Springer, Cham, 2018, pp. 116–131, doi: 10.1007/978-3-030-01264-9_8.
8. F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level Accuracy with 50x Fewer Parameters and <0.5MB Model Size," arXiv preprint arXiv:1602.07360, 2016.
9. K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More Features from Cheap Operations," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2020, pp. 1580–1589.
10. M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le, "MnasNet: Platform-Aware Neural Architecture Search for Mobile," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2019, pp. 2820–2828.
11. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.
12. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv preprint arXiv:2004.10934, 2020.
13. K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.
14. T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in Computer Vision – ECCV 2014, Lecture Notes in Computer Science, vol. 8693, Springer, Cham, 2014, pp. 740–755, doi: 10.1007/978-3-319-10602-1_48.
15. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," Int. J. Computer Vision, vol. 115, no. 3, pp. 211–252, 2015, doi: 10.1007/s11263-015-0816-y.
16. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2009, pp. 248–255.
17. M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes (VOC)

Challenge," Int. J. Computer Vision, vol. 88, no. 2, pp. 303–338, 2010.
18. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," Tech. Rep., University of Toronto, 2009.
19. G. Menghani, "Efficient Deep Learning: A Survey on Making Deep Learning Models Smaller, Faster, and Better," ACM Computing Surveys, vol. 55, no. 12, pp. 1–37, 2023.
20. P. Mittal, "A Comprehensive Survey of Deep Learning-Based Lightweight Object Detection Models for Edge Devices," Artificial Intelligence Review, vol. 57, art. 242, 2024.
21. T. Shahriar, "Comparative Analysis of Lightweight Deep Learning Models for Memory-Constrained Devices," arXiv preprint arXiv:2505.03303, 2025.
22. "Edge Deep Learning in Computer Vision and Medical Diagnostics: A Comprehensive Survey," Artificial Intelligence Review, Springer, 2025, doi: 10.1007/s10462-024-11033-5.
23. "Lightweight Deep Learning Models for Edge Devices — A Survey," International Journal of Computational Intelligence in Systems, 2024.
24. R. Banbury, V. J. Reddi, M. Lam, W. Fu, A. Fazel, J. Holleman, X. Huang, R. Hurtado, D. Kanter, A. Lokhmotov, D. Patterson, D. Pau, J.-s. Seo, J. Sieracki, U. Thakker, M. Verhelst, and P. Yadav, "Benchmarking TinyML Systems: Challenges and Direction," arXiv preprint arXiv:2003.04821, 2020.
25. "Efficient Edge Deployment of Quantized YOLOv4-Tiny for Aerial Emergency Object Detection on Raspberry Pi 5," arXiv preprint arXiv:2506.09300, 2025.
26. "Vision Transformers on the Edge: A Comprehensive Survey," Neurocomputing, 2025, arXiv preprint arXiv:2503.02891.
27. V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: On-device Real-time Body Pose Tracking," arXiv preprint arXiv:2006.10204, 2020.
28. T. Alatiah and C. Chen, "Recognizing Exercises and Counting Repetitions in Real Time," arXiv preprint arXiv:2005.03194, 2020.
29. "Deep Learning-Based Human Body Pose Estimation in Providing Feedback for Physical Movement: A Review," Heliyon / ScienceDirect, 2024 (PMC11401083).
30. Q. Li, Y. Liu, Y. Peng, C. Liu, J. Shi, F. Yan, and Q. Zhang, "Real-Time Facial Emotion Recognition Using Lightweight Convolution Neural Network," J. Phys.: Conf. Ser., vol. 1827, art. 012130, 2021, doi: 10.1088/1742-6596/1827/1/012130.
31. S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Do Better ImageNet Models Transfer Better?," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), 2019, pp. 2661–2671.
32. S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," in Proc. Int. Conf. Learning Representations (ICLR), 2016, arXiv preprint arXiv:1510.00149.
33. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2704–2713.
34. K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in Proc. Int. Conf. Learning Representations (ICLR), 2015, arXiv preprint arXiv:1409.1556.
35. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in Advances in Neural Information Processing Systems (NeurIPS), vol. 25, 2012, pp. 1097–1105.
36. Waqas, Muhammd, Muhammad Atif Tahir, and Rizwan Qureshi. "Ensemble-based instance relevance estimation in multiple-instance learning." In 2021 9th European workshop on visual information processing (EUVIP), pp. 1-6. IEEE, 2021.
37. Muhammad, N. A., Rehman, A., & Shoaib, U. (2017). Accuracy based feature ranking metric for multi-label text classification. International

Journal of Advanced Computer Science and Applications, 8(10).
38. Khan, M.A., Khan, S.U.R. & Lin, D. Shortening surgical time in high myopia treatment: a randomized controlled trial comparing non-OVD and OVD techniques in ICL implantation. BMC Ophthalmol 25, 303 (2025). https://doi.org/10.1186/s12886-025-04135-3
39. Shahzad, Inzamam, Jianquan Ouyang, and Saif Ur Rehman Khan. "FedVC-ADDiM: a federated learning framework for diagnosis of alzheimer disease using deep learning." Multimedia Systems 32, no. 3 (2026): 161. https://doi.org/10.1007/s00530-026-02229-6
40. Khan, S.U.R., Asif, S., Bilal, O. et al. Lead-cnn: lightweight enhanced dimension reduction convolutional neural network for brain tumor classification. Int. J. Mach. Learn. & Cyber. (2025). https://doi.org/10.1007/s13042-025-02637-6.
41. Khan, S.U.R., Zhao, M. & Li, Y. Detection of MRI brain tumor using residual skip block based modified MobileNet model. Cluster Comput 28, 248 (2025). https://doi.org/10.1007/s10586-024-04940-3
42. Hekmat, A., et al., Brain tumor diagnosis redefined: Leveraging image fusion for MRI enhancement classification. Biomedical Signal Processing and Control, 2025. 109: p. 108040.
43. Asim, M. N., Ibrahim, M. A., Malik, M. I., Dengel, A., & Ahmed, S. (2022). LGCA-VHPPI: A local-global residue context aware viral-host protein-protein interaction predictor. Plos one, 17(7), e0270275.
44. Ishfaque, Muhammad, Saif Ur Rehman Khan, and Yulong Lou. "Digitizing Health Monitoring in Engineering Structures Using Deep Learning: A Novel Block Architecture for Concrete Crack Prediction in Surface and Sub-surface Dataset." Journal of Bionic Engineering (2026): 1-23.
45. Khan, Saif Ur Rehman, Asif Raza, Inzamam Shahzad, and Ghazanfar Ali. "Enhancing concrete and pavement crack prediction through hierarchical feature integration with VGG16 and triple classifier ensemble." In 2024 Horizons of Information Technology and Engineering (HITE), pp. 1-6. IEEE, 2024.
46. Waqas, Muhammad, Zeshan Khan, Shaheer Anjum, and Muhammad Atif Tahir. "Lung-Wise Tuberculosis Analysis and Automatic CT Report Generation with Hybrid Feature and Ensemble Learning." In CLEF (Working notes), pp. 1-10. 2020.
47. Khan, S. U. R., & Khan, Z. (2025). Detection of Abnormal Cardiac Rhythms Using Feature Fusion Technique with Heart Sound Spectrograms. Journal of Bionic Engineering, 1-20.
48. Waqas, Muhammad, Syed Umaid Ahmed, Muhammad Atif Tahir, Jia Wu, and Rizwan Qureshi. "Exploring multiple instance learning (MIL): A brief survey." Expert Systems with Applications 250 (2024): 123893.
49. Al-Khasawneh, Mahmoud Ahmad, Asif Raza, Saif Ur Rehman Khan, and Zia Khan. "Stock Market Trend Prediction Using Deep Learning Approach." Computational Economics (2024): 1-32
50. Khan, Muhammad Ahmed, Manqiang Peng, Ding Lin, and Saif Ur Rehman Khan. "Deep Learning Based Estimation of Blood Glucose Levels from Multidirectional Scleral Blood Vessel Imaging." arXiv preprint arXiv:2603.12715 (2026).
51. Khan, Saif Ur Rehman, Muhammad Nabeel Asim, Sebastian Vollmer, and Andreas Dengel. "FloraSyntropy-net: scalable deep learning with novel FloraSyntropy archive for large-scale plant disease diagnosis." Plant Methods (2026).
52. Ur Rehman Khan, Saif, Omair Bilal, Arash Hekmat, Inzamam Shahzad, and Asif Raza. "Advancing food safety: deep learning for accurate detection of bacterial contaminants." Memetic Computing 18, no. 1 (2026): 11.
53. Waqas, Muhammad, Muhammad Atif Tahir, Sumaya Al-Maadeed, Ahmed Bouridane, and Jia Wu. "Simultaneous instance pooling and bag representation selection approach for multiple-instance learning (MIL) using vision transformer." Neural Computing and Applications 36, no. 12 (2024): 6659-6680.
54. Hekmat, Arash, Omair Bilal, Zuping Zhang, Saif Ur Rehman Khan, and Sohaib Asif.

"FRE-Net: A Fuzzy Richards Functions-Based Ensemble Network for Brain Tumor Detection." Journal of Bionic Engineering (2026): 1-23.
55. Mayumu, Nicanor, Xiaoheng Deng, Antoine Bagula, and Patrick Mukala. "V2X-JEPA: Self-Supervised Multi-Agent Joint Embedding Predictive Architecture for Robust Vehicle-to-Everything Perception." IEEE Internet of Things Journal (2026).
56. Waqas, Muhammad, Muhammad Atif Tahir, and Salman A. Khan. "Robust bag classification approach for multi-instance learning via subspace fuzzy clustering." Expert Systems with Applications 214 (2023): 119113.
57. Bilal, Omair, Arash Hekmat, Inzamam Shahzad, Asif Raza, and Saif Ur Rehman Khan. "Boosting Machine Learning Accuracy for Cardiac Disease Prediction: The Role of Advanced Feature Engineering and Model Optimization." The Review of Socionetwork Strategies (2025): 1-30.
58. Khan, Saif Ur Rehman, Muhammad Nabeel Asim, Sebastian Vollmer, and Andreas Dengel. "Temperature-driven robust disease detection in brain and gastrointestinal disorders via context-aware adaptive knowledge distillation." Biomedical Signal Processing and Control 112 (2026): 108671.
59. Khan, S. U. R., Asif, S., Zhao, M., Zou, W., Li, Y., & Xiao, C. (2026). ShallowMRI: A novel lightweight CNN with novel attention mechanism for Multi brain tumor classification in MRI images. Biomedical Signal Processing and Control, 111, 108425.
60. Khan, M. A., Khan, S. U. R., Rehman, H. U., Aladhadh, S., & Lin, D. (2025). Robust InceptionV3 with Novel EYENET Weights for Di-EYENET Ocular Surface Imaging Dataset: Integrating Chain Foraging and Cyclone Aging Techniques. International Journal of Computational Intelligence Systems, 18(1), 1-26.
61. Raza, Salahuddin, S. Latif, and G. Ali, “Refine Security Control Protocols for Block chain in Textile Industry Supply Chain Management”, VFAST trans. softw. eng., vol. 14, no. 2, pp. 62–80, Apr. 2026.
62. Khan, U. S., & Khan, S. U. R. (2025). Ethics by Design: A Lifecycle Framework for Trustworthy AI in Medical Imaging From Transparent Data Governance to Clinically Validated Deployment. arXiv preprint arXiv:2507.04249.
63. Waqas, Muhammad, Muhammad Atif Tahir, and Rizwan Qureshi. "Deep Gaussian mixture model based instance relevance estimation for multiple instance learning applications." Applied intelligence 53, no. 9 (2023): 10310-10325.
64. Khan, S. U. R., Rehman, H. U., & Bilal, O. (2025). AI-powered cancer diagnosis: classifying viable (live) vs non-viable (dead) cells using transfer learning. Signal, Image and Video Processing, 19(15), 1326.
65. Khan, S. U. R., Asif, S., Zhao, M., Zou, W., Li, Y., & Xiao, C. (2026). ShallowMRI: A novel lightweight CNN with novel attention mechanism for Multi brain tumor classification in MRI images. Biomedical Signal Processing and Control, 111, 108425.
66. Waqas M, Bandyopadhyay R, Showkatian E, Muneer A, Zafar A, Alvarez FR, Marin MC, Li W, Jaffray D, Haymaker C, Heymach J. The Next Layer: Augmenting Foundation Models with Structure-Preserving and Attention-Guided Learning for Local Patches to Global Context Awareness in Computational Pathology. arXiv preprint arXiv:2508.19914. 2025 Aug 27.
67. Shahzad, Inzamam, Asif Raza, and Muhammad Waqas. "Medical Image Retrieval using Hybrid Features and Advanced Computational Intelligence Techniques." Spectrum of engineering sciences 3, no. 1 (2025): 22-65.
68. Bilal, O., Hekmat, A., Shahzad, I. et al. Boosting Machine Learning Accuracy for Cardiac Disease Prediction: The Role of Advanced Feature Engineering and Model Optimization. Rev Socionetwork Strat (2025). https://doi.org/10.1007/s12626-025-00190-w
69. Asif Raza, Inzamam Shahzad, Ghazanfar Ali, and Muhammad Hanif Soomro. "Use Transfer Learning VGG16, Inception, and Reset50 to Classify IoT Challenge in Security

Domain via Dataset Bench Mark." Journal of Innovative Computing and Emerging Technologies 5, no. 1 (2025).
70. Khan, Z., Khan, S. U. R., Bilal, O., Raza, A., & Ali, G. (2025, February). Optimizing Cervical Lesion Detection Using Deep Learning with Particle Swarm Optimization. In 2025 6th International Conference on Advancements in Computational Sciences (ICACS) (pp. 1-7). IEEE.
71. Waqas, Muhammad, Zeshan Khan, Shaheer Anjum, and Muhammad Atif Tahir. "Lung-Wise Tuberculosis Analysis and Automatic CT Report Generation with Hybrid Feature and Ensemble Learning." In CLEF (Working notes), pp. 1-10. 2020.
72. Khan, S. U. R., Asif, S., Zhao, M., Zou, W., & Li, Y. (2025). Optimize brain tumor multiclass classification with manta ray foraging and improved residual block techniques. Multimedia Systems, 31(1), 1-27.
73. Asif Raza, Salahuddin, Ghazanfar Ali, Muhammad Hanif Soomro, Saima Batool, "Analyzing the Impact of Artificial Intelligence on Shaping Consumer Demand in E-Commerce: A Critical Review", International Journal of Information Engineering and Electronic Business(IJIEEB), Vol.17, No.5, pp. 42-61, 2025. DOI:10.5815/ijieeb.2025.05.04
74. Khan, M. A., Khan, S. U. R., Rehman, H. U., Aladhadh, S., & Lin, D. (2025). Robust InceptionV3 with Novel EYENET Weights for Di-EYENET Ocular Surface Imaging Dataset: Integrating Chain Foraging and Cyclone Aging Techniques. International Journal of Computational Intelligence Systems, 18(1), 204.
75. Nabeel Asim, M., Ali Ibrahim, M., Fazeel, A., Dengel, A., & Ahmed, S. (2023). Dna-mp: a generalized dna modifications predictor for multiple species based on powerful sequence encoding method. Briefings in Bioinformatics, 24(1), bbac546
76. Raza, Asif, Inzamam Shahzad, Muhammad Salahuddin, and Sadia Latif. "Satellite Imagery Employed to Analyze the Extent of Urban Land Transformation in The Punjab District of Pakistan." Journal of Palestine Ahliya University for Research and Studies 4, no. 2 (2025): 17-36.
77. Khan, S. U. R., Asif, S., Zhao, M., Zou, W., Li, Y., & Li, X. (2025). Optimized deep learning model for comprehensive medical image analysis across multiple modalities. Neurocomputing, 619, 129182.
78. Khan, Saif Ur Rehman, Asif Raza, Inzamam Shahzad, and Shehzad Khan. "Subcellular Structures Classification in Fluorescence Microscopic Images." In International Conference on Computing & Emerging Technologies, pp. 271-286. Cham: Springer Nature Switzerland, 2023.
79. Maqsood, H., & Khan, S. U. R. (2025). MeD-3D: A Multimodal Deep Learning Framework for Precise Recurrence Prediction in Clear Cell Renal Cell Carcinoma (ccRCC). arXiv preprint arXiv:2507.07839.
80. Raza, Asif, Salahuddin, & Inzamam Shahzad. (2024). Residual Learning Model-Based Classification of COVID-19 Using Chest Radiographs. Spectrum of Engineering Sciences, 2(3), 367–396.
81. Khan, S. U. R. (2025). Multi-level feature fusion network for kidney disease detection. Computers in Biology and Medicine, 191, 110214.
82. Salahuddin, Syed Shahid Abbas, Prince Hamza Shafique, Abdul Manan Razzaq, & Mohsin Ikhlaq. (2024). Enhancing Reliability and Sustainability of Green Communication in Next-Generation Wireless Systems through Energy Harvesting. Journal of Computing & Biomedical Informatics.
83. S. U. R. Khan, A. Raza, I. Shahzad and G. Ali, "Enhancing Concrete and Pavement Crack Prediction through Hierarchical Feature Integration with VGG16 and Triple Classifier Ensemble," 2024 Horizons of Information Technology and Engineering (HITE), Lahore, Pakistan, 2024, pp. 1-6.
84. Mahmood, F., Abbas, K., Raza, A., Khan,M.A., & Khan, P.W. (2019). Three Dimensional Agricultural Land Modeling using Unmanned Aerial System (UAS). International Journal of Advanced Computer Science and

Applications (IJACSA) [p-ISSN : 2158-107X, e-ISSN : 2156-5570], 10(1).
85. HUSSAIN, S., Raza, A., MEERAN, M. T., IJAZ, H. M., & JAMALI, S. (2020). Domain Ontology Based Similarity and Analysis in Higher Education. IEEEP New Horizons Journal, 102(1), 11-16.
86. Hekmat, A., Zuping, Z., Bilal, O., & Khan, S. U. R. (2025). Differential evolution-driven optimized ensemble network for brain tumor detection. International Journal of Machine Learning and Cybernetics, 1-26.
87. Raza, A., & Meeran, M. T. (2019). Routine of Encryption in Cognitive Radio Network. Mehran University Research Journal of Engineering and Technology [p-ISSN: 0254-7821, e-ISSN: 2413-7219], 38(3), 609-618.
88. Meeran, M. T., Raza, A., & Din, M. (2018). Advancement in GSM Network to Access Cloud Services. Pakistan Journal of Engineering, Technology & Science [ISSN: 2224-2333], 7(1).
89. Bilal, O., Hekmat, A., & Khan, S. U. R. (2025). Automated cervical cancer cell diagnosis via grid search-optimized multi-CNN ensemble networks. Network Modeling Analysis in Health Informatics and Bioinformatics, 14(1), 67.
90. Raza, Asif , Soomro, M. H., Shahzad, I., & Batool, S. (2024). Abstractive Text Summarization for Urdu Language. Journal of Computing & Biomedical Informatics, 7(02).
91. M. Wajid, M. K. Abid, A. Asif Raza, M. Haroon, and A. Q. Mudasar, “Flood Prediction System Using IOT & Artificial Neural Network”, VFAST trans. softw. eng., vol. 12, no. 1, pp. 210–224, Mar. 2024.
92. N. Mayumu, D. Xiaoheng, P. Mukala, S. U. R. Khan and M. U. Saeed, "Omni-V2X: A Vision-Language Model for Actionable Insights in Vehicle-to-Everything Systems," 2025 International Joint Conference on Neural Networks (IJCNN), Rome, Italy, 2025, pp. 1-8, doi: 10.1109/IJCNN64981.2025.11228491.
93. Waqas, M., Vierra, C., Kaplan, D. L., & Othman, S. (2019). Feasibility of low field MRI and proteomics for the analysis of tissue engineered bone. Biomedical Physics & Engineering Express, 5(2), 025037.
94. Khan, S. R., Asif Raza, Inzamam Shahzad, & Hafiz Muhammad Ijaz. (2024). Deep transfer CNNs models performance evaluation using unbalanced histopathological breast cancer dataset. Lahore Garrison University Research Journal of Computer Science and Information Technology, 8(1).
95. M. Waqas, Z. Khan, S. U. Ahmed and Asif. Raza, "MIL-Mixer: A Robust Bag Encoding Strategy for Multiple Instance Learning (MIL) using MLP-Mixer," 2023 18th International Conference on Emerging Technologies (ICET), Peshawar, Pakistan, 2023, pp. 22-26.
96. S. ur R. Khan, Asif. Raza, Muhammad Tanveer Meeran, and U. Bilhaj, “Enhancing Breast Cancer Detection through Thermal Imaging and Customized 2D CNN Classifiers”, VFAST trans. softw. eng., vol. 11, no. 4, pp. 80–92, Dec. 2023.
97. Yang, H., Khan, S. U. R., Bilal, O., Chen, C., & Zhao, M. (2025). CEOE-Net: Chaotic Evolution Algorithm-Based Optimized Ensemble Framework Enhanced with Dual-Attention for Alzheimer’s Diagnosis. *Computer Modeling in Engineering & Sciences*, *145*(2), 2401.
98. Chomba, B., Mukala, P., Mayumu, N., & Khan, S. U. R. (2025). DynaKG: Dynamic Knowledge Graph Attention With Learnable Temporal Decay for Recommendation. *IEEE Access*, *13*, 216956-216970.
99. O. Bilal, Asif Raza, S. ur R. Khan, and Ghazanfar Ali, “A Contemporary Secure Microservices Discovery Architecture with Service Tags for Smart City Infrastructures ”, VFAST trans. softw. eng., vol. 12, no. 1, pp. 79–92, Mar. 2024
100. S. U. R. Khan, A. Raza, I. Shahzad and G. Ali, "Enhancing Concrete and Pavement Crack Prediction through Hierarchical Feature Integration with VGG16 and Triple Classifier Ensemble," 2024 Horizons of Information Technology and Engineering (HITE), Lahore, Pakistan, 2024, pp. 1-6, doi: 10.1109/HITE63532.2024.10777242.

101. Ijaz, M., Khan, S. U. R., Rehman, A. U., Zaib, A., Vollmer, S., Dengel, A., & Asim, M. N. (2026). SolarFCD: A Large-Scale Dataset and Benchmark for Solar Fault Classification in Photovoltaic Systems. arXiv preprint arXiv:2604.23662.
102. Khan, S. U. R., Asim, M. N., Vollmer, S., & Dengel, A. (2026). FloraSyntropy-net: scalable deep learning with novel FloraSyntropy archive for large-scale plant disease diagnosis. Plant Methods.
103. Ishfaque, M., Khan, S. U. R., & Lou, Y. L. (2026). Towards efficient dam inspection: crack detection via chirplet transform feature and a pruned VGG16 architecture. *Memetic Computing*, *18*(1), 9.
104. S. Ur Rehman Khan, O. Bilal, S. Mistry, N. Deb, M. Mahmud and M. Bhuyan, "KDLight: A Lightweight Knowledge Distillation Framework for Medical Image Classification," *2025 International Joint Conference on Neural Networks (IJCNN)*, Rome, Italy, 2025, pp. 1-8, doi: 10.1109/IJCNN64981.2025.11228615.
105. O. Bilal, S. Ur Rehman Khan, S. Mistry, N. Deb, M. Mahmud and M. Bhuyan, "Towards Efficient Pruning and Multi-Scale Feature Transformations to Uncover Medical Diseases," *2025 International Joint Conference on Neural Networks (IJCNN)*, Rome, Italy, 2025, pp. 1-8, doi: 10.1109/IJCNN64981.2025.11229047.
106. Waqar, I. A., Zaib, A., Ahmed, S., Vollmer, S., Dengel, A., & Asim, M. N. (2026). MS-SSE-Net: A Multi-Scale Spatial Squeeze-and-Excitation Network for Structural Damage Detection in Civil and Geotechnical Engineering. arXiv preprint arXiv:2604.14711.
107. Khan, S. R., Raza, A., Waqas, M., & Raphay Zia, M. A. (2024). Efficient and Accurate Image Classification via Spatial Pyramid Matching and SURF Sparse Coding. Lahore Garrison University Research Journal of Computer Science and Information Technology, 7(4).
108. Stricker, Marco, Muhammad Nabeel Asim, Andreas Dengel, and Sheraz Ahmed. "CircNet: an encoder–decoder-based convolution neural network (CNN) for circular RNA identification." Neural Computing and Applications 34, no. 14 (2022): 11441-11452.
109. Shahzad, Inzamam, Asif Raza, Hasaan Maqsood, Saif Ur Rehman Khan, and Ghazanfar Ali. "Towards Robust Breast Cancer Diagnosis: A Hybrid Deep Learning Ensemble Framework." In 2025 Horizons of Information Technology and Engineering (HITE), pp. 1-6. IEEE, 2025.
110. Ijaz, M., Khan, S. U. R., Rehman, A. U., Asif, T., Vollmer, S., Dengel, A., & Asim, M. N. (2026). GlobalWasteData: A Large-Scale, Integrated Dataset for Robust Waste Classification and Environmental Monitoring. arXiv preprint arXiv:2602.07463.
111. Bilal, O., Hekmat, A., Khan, S. U. R., Raza, A., & Ali, G. (2025, December). MS-STO-Net: A Multi-Scale State Transition Optimization-Based Ensemble Network for Accurate White Blood Cell Classification. In *2025 27th International Multitopic Conference (INMIC)* (pp. 1-6). IEEE.
112. N. Mayumu, X. Deng, A. Bagula, S. u. R. Khan and P. Mukala, "V2X-JEPA: Self-Supervised Multi-Agent Joint Embedding Predictive Architecture for Robust Vehicle-to-Everything Perception," in *IEEE Internet of Things Journal*, doi: 10.1109/JIOT.2026.3660030.
113. Wasim, M., Mahmood, W., Asim, M. N., & Khan, M. U. (2018). Multi-label question classification for factoid and list type questions in biomedical question answering. IEEE Access, 7, 3882-3896.
114. Khan, S. U. R., Asim, M. N., Vollmer, S., & Dengel, A. (2025). Dynamic Weight Adjustment for Knowledge Distillation: Leveraging Vision Transformer for High-Accuracy Lung Cancer Detection and Real-Time Deployment. arXiv preprint arXiv:2510.20438.
115. Waqas, M., & Khan, M. A. (2018). JSOPT: A framework for optimization of JavaScript on web browsers. Mehran University Research Journal of Engineering & Technology, 37(1), 95-104.

116. Abbasi, A. F., Asim, M. N., Ahmed, S., Vollmer, S., & Dengel, A. (2024). Survival prediction landscape: an in-depth systematic literature review on activities, methods, tools, diseases, and databases. Frontiers in Artificial Intelligence, 7, 1428501.

IEER
Institute for Excellence in Education & Research