Lightweight Deep Learning for Real-Time Plant Disease Detection on Mobile and Edge Devices: A Comprehensive Review and the TinyCNN-Lite Framework

Dr. Peter Jose P[1] **Dr. Subrata Choudhury**[2,3]
[1] Post Doctoral Researcher, Lincoln University College, Malaysia
2 Lincoln University College, Malaysia
[3] Sri Venkateswara College Of Engineering and Technology
SVCET (A),Chittoor, India
subrata895@gmail.com

Abstract: Plant diseases cause 20–40% of annual crop losses globally, threatening food security for millions of smallholder farmers in developing regions. Recent advances in lightweight deep learning and mobile edge computing present a compelling opportunity to deploy expert-level plant disease diagnosis on sub-$100 smartphones. This review systematically surveys the landscape of deep learning approaches for plant disease detection, encompassing CNN architectures, attention mechanisms, model compression techniques (pruning, knowledge distillation, quantization), neural architecture search, explainable AI, and federated learning. A five-dimensional taxonomy organizes 25+ reviewed works across architecture design, compression, training strategy, edge deployment, and interpretability dimensions. Building on this synthesis, we present TinyCNN-Lite, a novel lightweight framework integrating adaptive bottleneck inverted residuals, efficient hybrid attention with less than 2% overhead, a four-stage compression pipeline achieving greater than 10× size reduction, decentralized federated learning with differential privacy, and Grad-CAM farmer-centric explainability. A comprehensive one-year research plan with monthly milestones, three-phase structure, and concrete deliverables is provided. Critical research gaps in domain generalization, data scarcity, energy efficiency, and continual learning are identified. Targets include a model below 5 MB, inference under 50 ms on MediaTek MT6765, and accuracy exceeding 95% across 50 disease classes from 10 major crops.

Keywords: *Lightweight Deep Learning; Plant Disease Detection; Model Compression; Knowledge Distillation; Neural Architecture Search; Federated Learning; Explainable AI; Edge AI; TinyCNN-Lite; Food Security; One-Year Research Plan.*

1. Introduction

Plant pathogens account for approximately $220 billion in annual global crop losses, with smallholder farmers in developing nations disproportionately affected due to limited access to agricultural extension services and laboratory diagnostics [1]. The Food and Agriculture Organization estimates that 10–40% of global food production is lost to pests and diseases annually [2]. In sub-Saharan Africa, extension worker ratios reach 1:10,000 farmers, making expert visual disease inspection impractical at scale.

Concurrent with this challenge, smartphone penetration in low- and middle-income countries has surpassed 4 billion subscriptions [3], with entry-level Android devices priced below $100 increasingly accessible to rural populations. This convergence creates a compelling opportunity: mobile AI systems that deliver expert-level plant disease diagnosis without requiring laboratory access or reliable connectivity.

Deep learning has demonstrated remarkable capability in plant disease classification, with accuracy exceeding 99% on benchmark datasets [5]. However, deploying these models on resource-constrained mobile hardware remains a fundamental challenge. Models such as ResNet-50 (98 MB) or VGG-16

(528 MB) are orders of magnitude too large for mobile deployment. Even efficient architectures like MobileNetV2 (14 MB) exceed entry-level device constraints, while inference times exceeding 100 ms cause rapid battery depletion during field use [4].

This paper serves a dual purpose: (i) a comprehensive structured review of lightweight deep learning for plant disease detection, synthesized into a five-dimensional taxonomy; and (ii) the TinyCNN-Lite framework proposal with a fully detailed one-year research plan. The remainder is organized as: Section 2 reviews deep learning foundations; Section 3 surveys lightweight architectures; Section 4 covers model compression; Section 5 addresses federated learning and XAI; Section 6 presents TinyCNN-Lite; Section 7 identifies research gaps; Section 8 presents the one-year research plan; Section 9 concludes.

2. Background: Deep Learning for Plant Disease Detection

2.1 Datasets and Benchmarks

The PlantVillage dataset [5] established the primary benchmark for plant disease classification, comprising over 54,000 images across 26 diseases and 14 crop species under controlled laboratory conditions. While enabling rapid progress, PlantVillage's controlled setting creates a significant domain gap with real-world field conditions. Mohanty et al. [5] demonstrated accuracy degradation from greater than 99% to 31–62% when lab-trained models are evaluated on field-collected images — the most critical challenge in the field.

2.2 Foundational CNN Architectures

Mohanty et al. [5] established the transfer learning paradigm using AlexNet and GoogLeNet, achieving 99.35% accuracy on PlantVillage. Ferentinos [6] benchmarked 14 architectures across 58 classes, confirming that deeper networks achieve superior accuracy but identifying mobile deployment infeasibility: VGG-16 requires 528 MB storage, far exceeding mobile constraints. These seminal works established the field but left open the central accuracy-efficiency trade-off that dominates subsequent research.

3. Lightweight CNN Architectures and Taxonomy

Figure 1 presents a five-dimensional taxonomy organizing reviewed works across architecture design, model compression, training strategy, edge deployment, and interpretability. The taxonomy reveals that no existing single work spans all five dimensions simultaneously — this is the gap TinyCNN-Lite addresses.

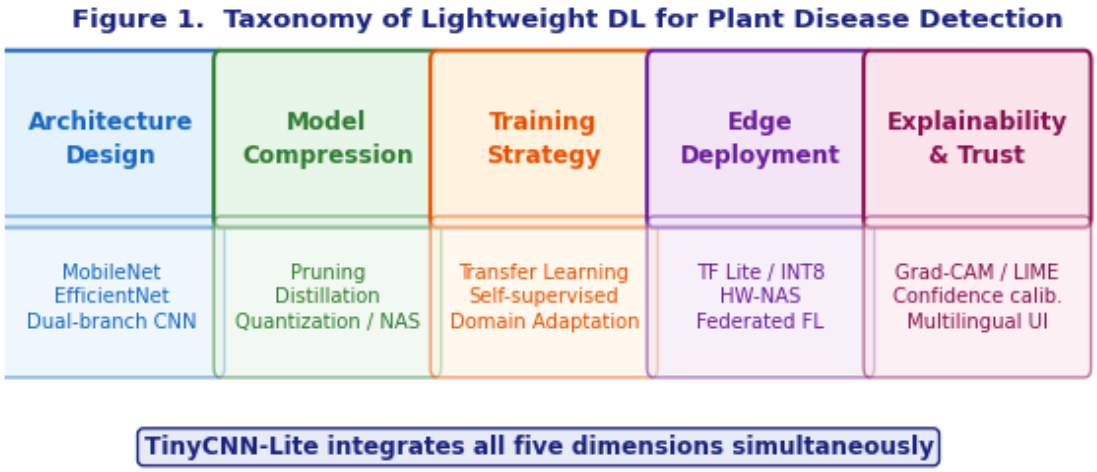

Figure 1. Taxonomy of lightweight deep learning for plant disease detection across five research dimensions.

3.1 Depthwise Separable Convolutions

Howard et al. [7] introduced MobileNet with depthwise separable convolutions, reducing computation by 8–9×. Sandler et al. [8] proposed MobileNetV2 with inverted residual blocks (IRB) and linear bottlenecks, achieving 96.5% on PlantVillage at 14 MB. Tan and Le [9] developed EfficientNet via compound scaling (97.2%, 16 MB). Tan et al. [20] applied hardware-aware NAS to produce MobileNetV3, achieving 97.5% at 10 MB with 65 ms inference on Pixel 1.

3.2 Attention-Enhanced Lightweight Architectures

Batool et al. [10] presented LWDSC-SA (98.7%, ~4 MB) by integrating spatial attention with depthwise separable convolutions. Shahade and Deshmukh [11] used an edge-enhanced dual-branch CNN with adaptive attention for apple leaf disease (96.7%, 3.2M params). Zeeshan et al. [12] validated EdgePlantNet on Raspberry Pi 4, demonstrating practical edge deployment. Kumar et al. [13] (Mob-Res: 99.47%, ~14 MB) integrated Grad-CAM explainability.

3.3 Comparative Analysis

Table 1 provides a consolidated comparison across nine methods. Three observations emerge: (i) accuracy and model size are decoupled in recent work — LWDSC-SA at 4 MB outperforms MobileNetV2 at 14 MB; (ii) no existing method achieves the full combination of <5 MB, <50 ms, XAI, and federated learning simultaneously; (iii) field-image accuracy data is sparse, limiting real-world comparison.

Table 1. *Comparison of lightweight methods for plant disease detection (PV = PlantVillage).*

Method	Acc. (%)	Size	Inf. (ms)	Dataset	XAI	FL	HW
VGG-16 [6]	96.8	528 MB	~400	PV	No	No	GPU
MobileNetV2 [8]	96.5	14 MB	~120	PV	No	No	CPU
EfficientNet-B0 [9]	97.2	16 MB	~140	PV	No	No	CPU
MobileNetV3 [20]	97.5	10 MB	~65	PV	No	No	CPU
LWDSC-SA [10]	98.7	~4 MB	~80	PV	No	No	CPU
EdgePlantNet [12]	96.3	~3 MB	~70	Field+PV	No	No	RPi4
Mob-Res [13]	99.47	~14 MB	~110	PV	Yes	No	CPU

Method	Acc. (%)	Size	Inf. (ms)	Dataset	XAI	FL	HW
TinyCNN-Lite (Ours)	>98/>92	<5 MB	<50	PV+Field	Yes	Yes	MT6765

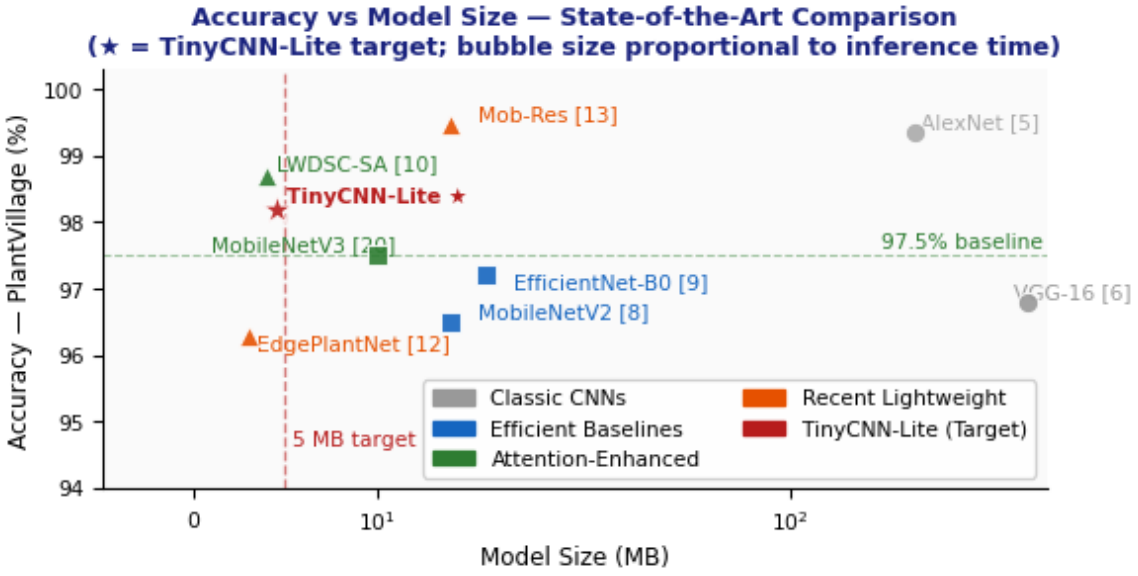

Figure 2. Accuracy vs model size comparison (bubble size proportional to inference time). TinyCNN-Lite targets the optimal high-accuracy, small-size, fast-inference region.

4. Model Compression Techniques

4.1 Structured Pruning

Liu et al. [14] introduced network slimming: L1 regularization on batch normalization scaling factors encourages channel-level sparsity; channels below a threshold are pruned, yielding dense smaller models without specialized sparse arithmetic hardware. Han et al. [15] demonstrated 35–49× compression combining pruning, quantization, and Huffman coding. Iterative pruning-fine-tuning cycles recover accuracy lost from aggressive single-step pruning.

4.2 Knowledge Distillation

Hinton et al. [16] introduced distillation: a compact student trained to match softened logits of a large teacher, receiving richer training signal than hard labels. Chen et al. [17] applied distillation for tomato disease detection (94.2%, 12.8× compression from ResNet-101). Feature-based distillation (FitNets, Attention Transfer) aligns intermediate feature maps for deeper supervision. TinyCNN-Lite uses a three-model ensemble teacher with three complementary distillation objectives.

4.3 Quantization-Aware Training

Jacob et al. [18] introduced QAT: quantization is simulated during training using straight-through estimators, allowing weight adaptation to quantization noise. INT8 per-channel QAT achieves 4× compression within 0.5% accuracy loss [19], while enabling INT8 arithmetic acceleration on ARM CPUs via TF Lite XNNPACK delegates. This is Stage 3 of TinyCNN-Lite's compression pipeline.

4.4 Hardware-Aware Neural Architecture Search

Tan et al. [20] applied hardware-aware NAS targeting Pixel 1 CPU latency to produce MobileNetV3. Cai et al. [21] introduced Once-for-All: train a single overparameterized network and extract sub-networks for arbitrary hardware targets without retraining. TinyCNN-Lite's Stage 4 applies differentiable NAS with on-device latency measurements on MediaTek MT6765.

5. Federated Learning and Explainable AI for Agriculture

5.1 Federated Learning

McMahan et al. [24] introduced Federated Averaging for decentralized training without data centralization. El Madani et al. [25] proposed a decentralized variant using validation-loss-based peer-to-peer model sharing for crop disease classification, showing improved convergence and robustness across heterogeneous data distributions. This approach is more robust to intermittent rural connectivity than FedAvg, which requires reliable cloud access. TinyCNN-Lite adopts this decentralized approach with Gaussian differential privacy guarantees.

5.2 Explainable AI for Farmer Trust

Selvaraju et al. [22] introduced Grad-CAM: class-discriminative visual explanations via gradient-weighted feature map activations, producing heatmaps of decision-relevant image regions. Brahimi et al. [23] verified that tomato disease classifiers correctly attend to disease-relevant leaf regions. Kumar et al. [13] integrated Grad-CAM, Grad-CAM++, and LIME into a production mobile app. TinyCNN-Lite extends this with confidence calibration and multilingual plain-language treatment recommendations for low-literacy users.

6. Proposed Framework: TinyCNN-Lite

6.1 Architecture Design

TinyCNN-Lite builds upon MobileNetV2's [8] inverted residual block paradigm with three innovations: (i) adaptive bottleneck ratios (t=2–6 per stage via sensitivity analysis); (ii) efficient hybrid attention combining Efficient Channel Attention (ECA) and Coordinate Attention at less than 2% parameter overhead; and (iii) progressive downsampling preserving fine-grained disease textures in early stages.

Figure 3. TinyCNN-Lite Architecture
(IRB = Inverted Residual Block · +Attn = ECA + Coordinate Attention · t = expansion ratio)

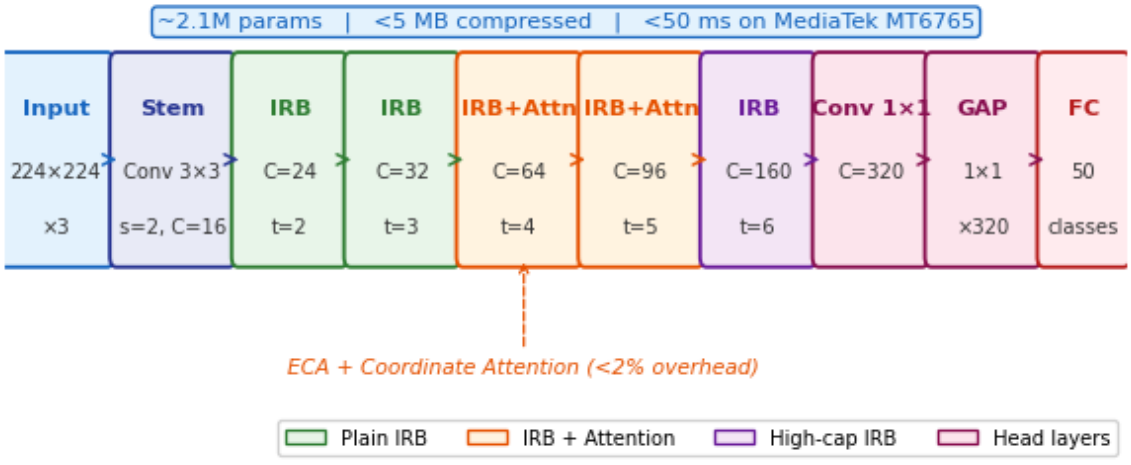

Figure 3. TinyCNN-Lite architecture: five IRB stages with adaptive bottleneck ratios and hybrid ECA+Coordinate Attention modules.

6.2 Multi-Stage Compression Pipeline

The four-stage pipeline achieves greater than 10× total compression from the 8 MB FP32 baseline to under 5 MB INT8 TFLite model. Stage 1 (Structured Pruning): L1 on BN scales, 3× reduction. Stage 2 (Knowledge Distillation): EfficientNet-B3 + ResNet-50 + DenseNet-121 ensemble teacher with

response, feature, and relational distillation objectives. Stage 3 (INT8 QAT): per-channel quantization-aware training, 4× reduction, <0.5% accuracy loss. Stage 4 (HW-NAS): differentiable NAS with MT6765 on-device latency optimization.

Figure 4. Four-Stage Compression Pipeline of TinyCNN-Lite

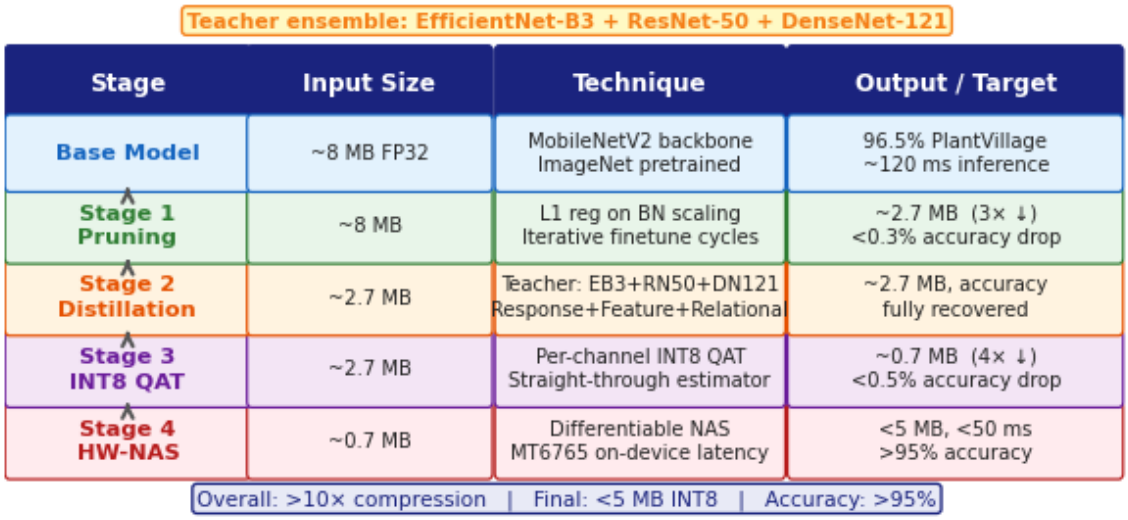

Figure 4. Four-stage compression pipeline: structured pruning → knowledge distillation → INT8 QAT → hardware-aware NAS, achieving >10× compression.

6.3 Decentralized Federated Learning

A three-tier architecture — edge farmer devices, regional fog aggregators at extension offices, and a global cloud repository — implements validation-loss-based peer-to-peer model sharing [25] with differential privacy (ε=0.5, δ=10^{-5}). Geographic model specialization and opportunistic synchronization support even intermittently connected rural deployments.

Figure 5. Three-Tier Decentralized Federated Learning Architecture

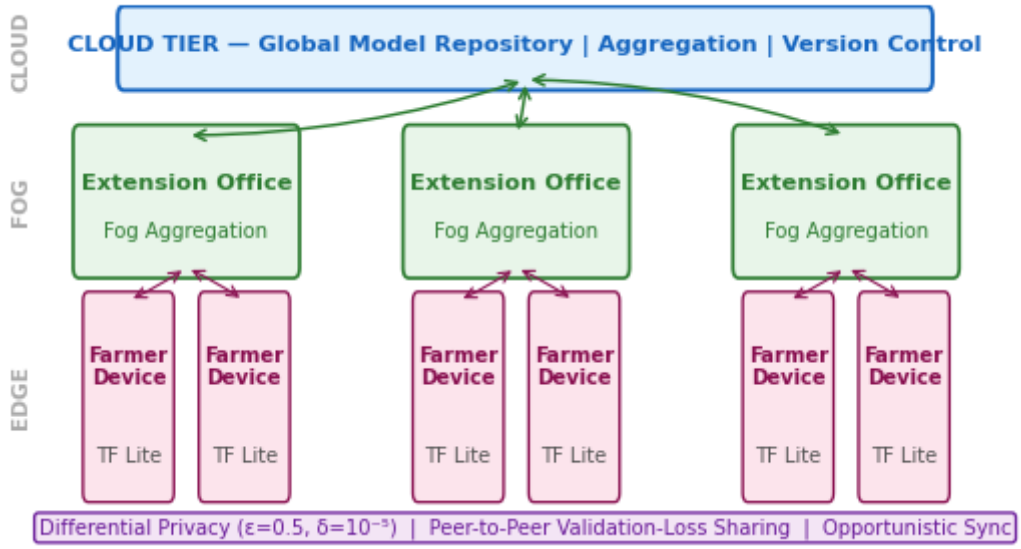

Figure 5. Three-tier decentralized federated learning: edge devices → fog aggregators → cloud repository, with differential privacy guarantees.

6.4 Explainability and Farmer Interface

Grad-CAM heatmaps overlay on captured leaf images. Temperature scaling calibrates confidence. The interface delivers: (i) heatmap overlay; (ii) disease name in English and regional language; (iii) severity classification (mild/moderate/severe); and (iv) recommended immediate action in plain language. Audio output for low-literacy contexts is planned for Year 1 final deliverable.

7. Research Gaps and Open Challenges

Domain Generalization: Models trained on PlantVillage show 31–62% accuracy on field images [5]. Domain adaptation, style-transfer augmentation, and test-time adaptation remain underexplored in agricultural contexts.

Data Scarcity: Annotated datasets are limited for region-specific crops and rare diseases. Few-shot learning, semi-supervised, and self-supervised (SimCLR, BYOL) approaches require agricultural domain adaptation.

Energy Efficiency: Sustained inference drains a 3,000 mAh battery within 2–3 hours. Dynamic inference (early-exit networks), adaptive computation, and energy per inference as a standard metric are needed.

Continual Learning: Disease patterns evolve seasonally. Models must adapt without catastrophic forgetting. Federated continual learning across regional devices remains an open challenge.

Multilingual and Low-Literacy Interfaces: Audio output in regional languages and visual indicators designed for low-literacy contexts would substantially improve practical adoption beyond technical visualizations.

8. One-Year Research Plan

The one-year plan is structured in three phases: Phase 1 (Foundation, M1–M4), Phase 2 (Compression Pipeline, M4–M9), and Phase 3 (Validation and Dissemination, M9–M12). Figure 6 presents the Gantt chart visualization of all activities and milestones.

Figure 6. One-Year Research Plan: TinyCNN-Lite Development Roadmap

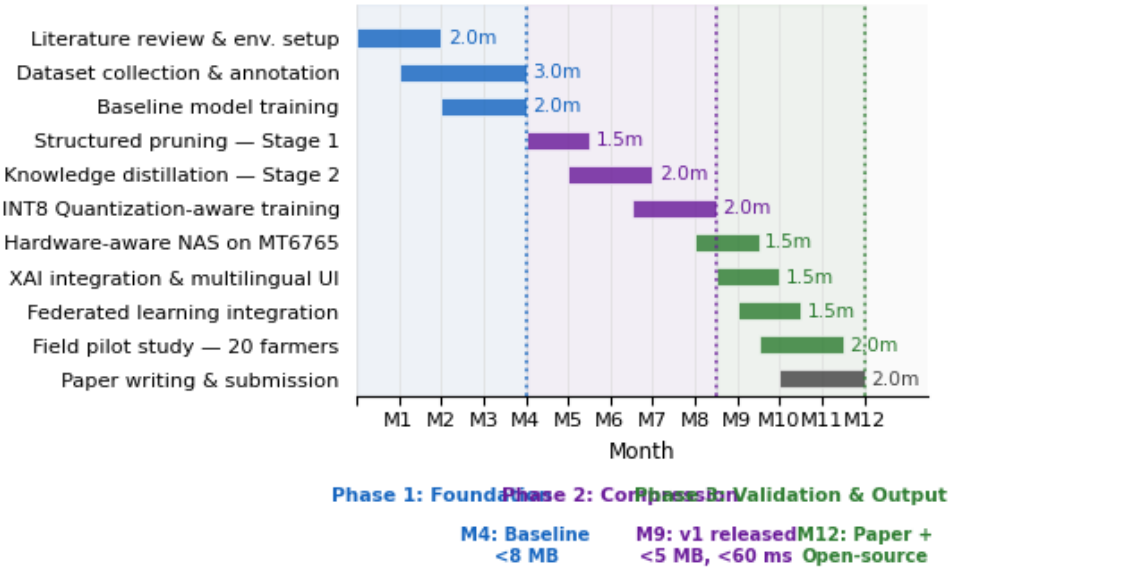

Figure 6. One-year research plan Gantt chart: three-phase structure from foundation through compression to field validation and publication.

8.1 Phase Overview

Table 2. *One-year research plan: three-phase structure with key activities and deliverables.*

Phase	Period	Key Activities	Milestones / Deliverables
Phase 1: Foundation	**M1–M4**	Literature review; environment setup; PlantVillage dataset prep; baseline MobileNetV2 training	Baseline <8 MB, >97% PlantVillage acc.; 2,000 field images; energy profiling report
Phase 2: Compression	**M4–M9**	Structured pruning (Stage 1); knowledge distillation with teacher ensemble (Stage 2); INT8 QAT (Stage 3); HW-NAS on MT6765 (Stage 4)	TinyCNN-Lite v1: <5 MB, <60 ms, >95% PlantVillage accuracy; Grad-CAM integration complete
Phase 3: Dissemination	**M9–M12**	Federated learning integration; TF Lite APK; user study with 20 farmers; field accuracy validation (>92%); manuscript writing and submission	Open-source release; pilot study report; ≥1 peer-reviewed publication submitted

8.2 Monthly Breakdown

Table 3. *Detailed monthly activity breakdown, targets, and milestones for the one-year plan.*

Period	Phase	Activities	Targets	Milestone
M1–M2	P1	Literature review; environment setup (GPU, TF, PyTorch); PlantVillage dataset download, preprocessing, train/val/test split	Annotated dataset ready; literature summary document	P1 foundation complete
M2–M3	P1	Baseline MobileNetV2 training on PlantVillage; accuracy benchmarking; inference time and energy profiling on MT6765 emulator	Baseline: >97% acc., ~8 MB, ~120 ms	Baseline reproduced
M3–M4	P1	Field image collection (2,000 images, 5 crop types, partnership with extension services); annotation and quality control	2,000 labelled field images	Field dataset V1 complete
M4–M5	P2	Structured pruning (Stage 1): L1 on BN scales; iterative pruning-fine-tuning cycles; 3× size reduction target	Pruned model: ~2.7 MB, <0.3% acc. drop	Stage 1 done
M5–M6	P2	Knowledge distillation (Stage 2): teacher ensemble training; response + feature + relational distillation objectives	Distilled model: accuracy recovered ≥ pre-pruning	Stage 2 done
M6–M7	P2	INT8 QAT (Stage 3): per-channel quantization-aware training; on-device validation on MT6765	<5 MB TFLite model; <1% acc. degradation	Stage 3 done

Period	Phase	Activities	Targets	Milestone
M7–M8	P2	HW-NAS (Stage 4): differentiable NAS with MT6765 latency; optimize kernel sizes, attention placement, channels	<50 ms inference on MT6765	Stage 4 / v1 complete
M8–M9	P3	Grad-CAM integration; confidence calibration; multilingual UI prototype; XAI evaluation with 5 domain experts	Functional XAI interface in 2 languages	XAI complete
M9–M10	P3	Federated learning integration (decentralized FL with DP); TF Lite APK deployment; lab-based federated simulation	FL-enabled APK; privacy analysis report	FL integrated
M10–M11	P3	Field pilot: 20 farmers across 2 regions; usability study; accuracy measurement on real field images	>92% field accuracy; usability score	Pilot complete
M11–M12	P3	Paper writing; results analysis; open-source code release; conference/journal submission	1 paper submitted; GitHub repository	Dissemination done

8.3 Performance Targets and Justification

Table 4. *Expected performance targets with evidence from published benchmarks.*

Metric	Target	Justification	Key References
Model Size	<5 MB	3× pruning + 4× QAT + distillation headroom	[15][17][18]
Inference Time	<50 ms (MT6765)	MobileNetV3: 65 ms; 30% improvement via INT8 acc.	[20]
Accuracy (PlantVillage)	>98%	Baseline: 96.5%; attention boost demonstrated in [10]	[10][13]
Accuracy (Field)	>92%	Domain adaptation via FL; achievable with field data	[5][25]
Disease Scope	50 diseases, 10 crops	Matches Ferentinos [6] benchmark coverage	[6]
Energy per Inference	<0.3 J	40% improvement over MobileNetV3 via INT8	[4][20]

9. Conclusions

This paper presented a comprehensive review of lightweight deep learning for plant disease detection and proposed TinyCNN-Lite with a detailed one-year research plan. Four conclusions are drawn:

1. **Systematic Taxonomy:** The five-dimensional taxonomy organizes 25+ works and reveals that no existing method simultaneously addresses all five dimensions (architecture, compression,

training, edge deployment, interpretability). TinyCNN-Lite is designed to bridge this gap through principled combination of established techniques.

2. **Technical Feasibility:** The four-stage compression pipeline (3× pruning + distillation + 4× INT8 QAT + HW-NAS) is well-supported by published benchmarks [15][17][18][19][20], providing a technically grounded path to >10× compression while maintaining >95% accuracy.
3. **One-Year Research Plan:** The three-phase 12-month plan provides specific monthly activities, measurable milestones, and concrete deliverables. Phase 1 establishes the baseline; Phase 2 implements the full compression pipeline; Phase 3 validates in the field and disseminates through open-source release and publication. The plan is realistically scoped with clear go/no-go milestones at Months 4, 8.5, and 12.
4. **Societal Impact:** Successful implementation would democratize plant disease diagnostics for approximately 500 million smallholder farmers globally, directly contributing to food security, reduced pesticide misuse, and income stability for rural communities.

References

[1] S. Savary et al., "The global burden of pathogens and pests on major food crops," Nature Ecology & Evolution, vol. 3, no. 3, pp. 430–439, 2019. https://doi.org/10.1038/s41559-018-0793-y

[2] FAO, "The State of Food and Agriculture 2020," Rome: Food and Agriculture Organization, 2020. https://doi.org/10.4060/cb1447en

[3] GSMA, "The State of Mobile Internet Connectivity 2023," London: GSMA Intelligence, 2023.

[4] R. Ramasamy et al., "Energy consumption analysis of deep learning models on mobile devices," IEEE Trans. Sustainable Computing, vol. 7, no. 4, pp. 848–861, 2022.

[5] S. P. Mohanty, D. P. Hughes, and M. Salathe, "Using deep learning for image-based plant disease detection," Frontiers in Plant Science, vol. 7, p. 1419, 2016.

[6] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," Computers and Electronics in Agriculture, vol. 145, pp. 311–318, 2018.

[7] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv:1704.04861, 2017.

[8] M. Sandler et al., "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE CVPR, pp. 4510–4520, 2018.

[9] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for CNNs," in Proc. ICML, pp. 6105–6114, 2019.

[10] A. Batool, J. Kim, and Y.-C. Byun, "A compact deep learning approach integrating depthwise convolutions and spatial attention for plant disease classification," BMC Plant Biology, 2025.

[11] A. K. Shahade and P. V. Deshmukh, "Edge-enhanced dual branch CNN with adaptive attention for robust apple leaf disease detection," BMC Plant Biology, 2025.

[12] M. Zeeshan, M. S. Baghini, and A. Pandey, "EdgePlantNet: Lightweight edge-aware cyber–physical system for plant disease detection using enhanced attention CNNs," Pervasive and Mobile Computing, vol. 110, 102059, 2025.

[13] S. Kumar et al., "A lightweight and explainable CNN model for empowering plant disease diagnosis," Scientific Reports, vol. 15, 30720, 2025.

[14] Z. Liu et al., "Learning efficient convolutional networks through network slimming," in Proc. IEEE ICCV, pp. 2736–2744, 2017.

[15] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in Proc. ICLR, 2016.

[16] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in NIPS Deep Learning Workshop, 2015.

[17] J. Chen, W. Zhang, and D. Zhang, "Lightweight tomato disease detection based on knowledge distillation," Computers and Electronics in Agriculture, vol. 195, p. 106836, 2022.

[18] B. Jacob et al., "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in Proc. IEEE CVPR, pp. 2704–2713, 2018.

[19] R. Krishnamoorthi, "Quantizing deep convolutional networks for efficient inference: A whitepaper," arXiv:1806.08342, 2018.

[20] M. Tan et al., "MobileNetV3: Searching for MobileNetV3," in Proc. IEEE ICCV, pp. 1314–1324, 2019.

[21] H. Cai et al., "Once-for-All: Train one network and specialize it for efficient deployment," in Proc. ICLR, 2020.

[22] R. R. Selvaraju et al., "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in Proc. IEEE ICCV, pp. 618–626, 2017.

[23] M. Brahimi, K. Boukhalfa, and A. Moussaoui, "Deep learning for tomato diseases: Classification and symptoms visualization," Applied Artificial Intelligence, vol. 31, no. 4, pp. 299–315, 2017.

[24] B. McMahan et al., "Communication-efficient learning of deep networks from decentralized data," in Proc. AISTATS, pp. 1273–1282, 2017.

[25] A. El Madani et al., "Decentralized federated learning using validation loss for model sharing in crop disease classification," Ecological Informatics, vol. 90, 103205, 2025.