BoltNet: An Ultra-Lightweight Convolutional Network for On-Device Plant Species Identification

Daniel Rossi, Guido Borghi, and Roberto Vezzani

University of Modena and Reggio Emilia, Modena, Italy
{name.surname}@unimore.it

Abstract. Automated plant species identification from citizen-science imagery is an established, demanding fine-grained recognition problem: large taxonomic label spaces, visually similar species, and long-tailed observations require real model capacity, while field use constrains memory, latency, and power. Model size is only part of the deployment cost: intermediate activations held in memory during inference and platform-dependent execution behavior matter too, so compact recognition must be assessed on target hardware rather than through complexity metrics alone. We present BoltNet, an ultra-lightweight fully convolutional architecture combining a Spatial Redistribution Bottleneck and Logit Pre-Sampling to improve the tradeoff between predictive performance and model size in high-cardinality classification, and report the Accuracy-Compression Tradeoff as a complementary diagnostic. On Pl@ntNet-300K, BoltNet reaches 0.682 F1-score with 341K parameters (1.37 MB), the highest F1-score among evaluated models below 2 MB and close to substantially larger convolutional backbones. Model-only measurements on a Raspberry Pi 5, Jetson Orin Nano, and Hailo-8 characterize execution across CPU, GPU, and NPU platforms, where BoltNet is the most consistently efficient model, with the best FPS/W on the GPU and NPU and second-best on the CPU. Results on AIDERv2 and CLRS provide secondary evidence of transfer across environmental image-classification tasks. Code available at: https://codeberg.org/danielrossi/BoltNet.

Keywords: Plant species identification · Ultra-lightweight CNN · Edge inference · Fine-grained classification · Energy efficiency

1 Introduction

Identifying plant species from field photographs underpins a growing set of ecological and agricultural applications, from citizen-science biodiversity monitoring to crop and phenotyping pipelines that increasingly rely on image analysis [4, 11, 24]. The problem is well studied yet far from solved: systematic reviews describe automated species identification as a recognition task whose difficulty stems from the acquisition process itself, since the visible organ, the phenological stage, the viewpoint, and the image composition each change the evidence available to a classifier [28,30,37,38]. It is fine-grained in the strict sense, with related

species differing by subtle traits while images of one species vary widely, which separates it from the balanced, low-cardinality benchmarks on which compact models are usually tuned.

Citizen-science platforms have turned this into a large-scale recognition problem. Pl@ntNet-300K assembles more than 300,000 observations across 1,081 species with high label ambiguity and a steep long-tailed distribution, the least represented 80% of species accounting for only 11% of the images [6], and crowdsourced records of this kind now reach species and regions that conventional surveys cannot cover [18]. The long tail is not a mere statistical nuisance: rare taxa are often the most consequential to recognize, as in the early detection of invasive or emerging threats [20], and they are exactly where compact models tend to fail. The practical relevance of accurate yet small models is reinforced by evidence that freely available identification apps already reach useful accuracy for everyday users [27].

Running this recognition in the field changes the binding constraint from accuracy alone to the resources of the device that carries the model, where memory, latency, and power are all limited. Model size is only one part of that cost. At inference, the peak working memory is governed by the intermediate activation tensors, which in standard convolutional networks concentrate in the early high-resolution stages and can exceed the weight footprint [17], while parameter count and FLOPS are themselves weak predictors of latency and energy once data movement and operator support are accounted for [19, 34]. A model is deployable only when its weights, its activation footprint, and its operators jointly fit the target, which is why a dedicated family of ultra-lightweight networks such as EmergencyNet and TakuNet has emerged for the tightest on-device budgets, roughly an order of magnitude smaller than common lightweight and mobile backbones [13, 25]. We address this ultra-lightweight regime, which we delimit by a model size below 2 MB and a budget below 100 M FLOPS, and we benchmark against representatives of both the established lightweight backbones and the dedicated ultra-lightweight models, so that the comparison spans the spectrum rather than a single side of it.

Within this setting we present BoltNet, an ultra-lightweight fully convolutional architecture built on two parameter-free rearrangements: a Spatial Redistribution Bottleneck in the backbone and Logit Pre-Sampling before the classifier, which together lower parameter cost while keeping dense, hardware-friendly operators (Sect. 3). On Pl@ntNet-300K, BoltNet is the most accurate model below 2 MB and stays close to substantially larger convolutional backbones, and across a CPU, a GPU, and an NPU it is the most consistently efficient of the models we compare. AIDERv2 [33] and CLRS [14] serve as secondary transfer checks, and we report the Accuracy-Compression Tradeoff (ACT) as a complementary diagnostic of accuracy retention against parameter reduction.

The contributions of this work are:

1. We introduce BoltNet, an ultra-lightweight convolutional architecture that combines the Spatial Redistribution Bottleneck and Logit Pre-Sampling to improve the recognition-to-size tradeoff in high-cardinality classification.

2. We evaluate BoltNet on Pl@ntNet-300K, where it attains the highest F_1-score among the models below 2 MB while remaining competitive with substantially larger convolutional backbones, with secondary transfer checks on AIDERv2 and CLRS.
3. We report model-only execution across CPU, GPU, and NPU platforms, where BoltNet achieves the highest FPS/W on the Jetson Orin Nano and Hailo-8 and the second-highest on the Raspberry Pi 5.
4. We report ACT as a complementary metric that jointly summarizes accuracy retention and parameter reduction across architectural variants.

2 Related Work

Our work sits at the intersection of two research directions that are usually pursued separately: image-based plant species identification under realistic acquisition conditions, and efficient visual recognition on embedded hardware. The first defines the recognition challenge, including fine-grained visual differences, heterogeneous observations, and large, imbalanced taxonomic spaces. The second concerns the architectural and system constraints that decide whether a model can run within limited memory, latency, and power budgets. We review the two directions before positioning BoltNet among deployment-aware compact architectures.

2.1 Plant identification in realistic conditions

Image-based plant identification has moved from curated specimens and isolated organs toward photographs taken by non-experts in the field, a transition documented since the earliest social-image systems [11, 38]. This change turned the acquisition process into part of the recognition problem: depending on the observation, a classifier may receive a flower, a leaf, a fruit, a partial view, or a cluttered scene in which only a few diagnostic traits are visible. Comparative studies show that the visible organ and the image composition materially affect which cues are available, so this variability is intrinsic to the data rather than an artefact of any single model [28, 30]. The problem is also taxonomic in nature: recognition can be assessed at genus or family level, and accuracy drops for unseen or rarely observed species [31, 37], a regime that tends to penalize low-capacity models.

Citizen-science platforms scaled these difficulties by orders of magnitude. Pl@ntNet-300K combines a 1,081-class taxonomic space with strong label ambiguity and a steep long-tailed distribution, and is demanding for reasons that go beyond image count [6]; the structured collection of field observations that feeds such platforms is itself an active subject of study [1]. Fine-grained plant recognition has accordingly been treated as a problem in its own right, often through specialized or transfer-learned networks [39], and recent evidence that free identification apps can reach high accuracy in everyday use makes the deployability of these models a concrete concern rather than a benchmark abstraction [27].

Species identification is not equivalent to plant phenotyping, but the two share field imagery, sensing platforms, and computational infrastructure, so an efficient recognition backbone can support broader plant-analysis pipelines without itself constituting a complete phenotyping solution [12].

2.2 Embedded inference for environmental vision

Field-deployed vision operates under constraints that differ from those of server-side recognition. Acquisition, inference, communication, and application logic share a single memory and power budget, and network weights must remain resident throughout execution, which is why work on memory-constrained deployment has treated model storage as a primary design variable [3, 26]. Storage is not the only memory cost. At inference, the peak working memory is set by the largest intermediate activations, which standard convolutional designs concentrate in the early, high-resolution stages and which can dominate the weights [17], while the final linear layer adds a further, class-dependent burden that grows with the taxonomic space. Reducing weights therefore does not by itself bound the memory an inference allocates, but it does leave headroom for the activation working set and for the rest of the pipeline.

Deployment cost is likewise only partly captured by parameter count and FLOPS. Data movement, the memory hierarchy, tensor shapes, kernel implementation, and runtime scheduling all influence latency and energy [34], and ShuffleNetV2 showed that networks with similar arithmetic complexity can differ sharply in execution time because memory-access cost and parallelism depend on the architecture and the platform [19]. Deployment frameworks such as DORY make the same point from the systems side, where practical execution hinges on compilation, tiling, and DMA scheduling [2]. These effects matter most when a single model is expected to run across CPUs, embedded GPUs, and dedicated NPUs, each exposing a different balance of parallelism, bandwidth, and operator support, so direct measurement on the target devices is needed to complement size and FLOPS. Compact environmental classifiers such as EmergencyNet and TakuNet show that useful recognition is possible within very small budgets, yet their primary tasks contain few classes and do not impose the fine-grained burden of a benchmark such as Pl@ntNet-300K [13, 25].

2.3 Embedded and ultra-lightweight architectures

Mobile CNN research has produced a sequence of strategies for trading cost against accuracy. MobileNet introduced depthwise-separable convolutions and then the inverted residual bottleneck, in which channel expansion and linear projection balance capacity and computation [10,29]; MobileNetV3 added hardware-aware search and measured latency [9]; GhostNet reduced feature redundancy through cheap linear operations [7]; ShuffleNetV2 foregrounded practical execution characteristics; and EfficientNet studied the coordinated scaling of depth, width, and input resolution [19,35,36]. More recent compact models incorporate

attention: MobileViT and its separable-attention variant combine local convolution with global context, and EfficientFormer adapts transformer-style processing to latency-constrained deployment [16, 21, 22]. These hybrids are strong baselines rather than a foil: attention is not impractical in principle, but its operators tend to map less favorably onto the embedded runtimes considered here, so a low parameter count does not by itself identify the best deployment point, as our measurements confirm (Sect. 5).

Taken together, the two sides of the problem are usually studied apart. Plant-identification research examines realistic observations, large taxonomic spaces, and long-tailed data, but rarely treats heterogeneous embedded execution as a primary objective; conversely, ultra-lightweight environmental models and generic mobile backbones are typically evaluated on lower-cardinality tasks or on a narrow set of devices. BoltNet is positioned at this intersection. It extends the inverted residual design with the Spatial Redistribution Bottleneck, which trades channels for spatial resolution through a lossless rearrangement and so lowers the backbone's parameter cost while keeping dense kernels, in contrast to efficiency obtained through grouped, atrous, or otherwise sparse operators. Logit Pre-Sampling applies a further feature tensor shape reorganization to contain the classifier in high-cardinality settings. Pl@ntNet-300K tests whether this design preserves fine-grained recognition within an ultra-lightweight budget, measurements on CPU, GPU, and NPU assess whether the resulting operating point holds across heterogeneous runtimes, and AIDERv2 and CLRS serve as secondary checks on other environmental image-classification tasks.

3 BoltNet

BoltNet is a small, fully convolutional network for real-time inference on constrained edge hardware. It rests on two components: the Spatial Redistribution Bottleneck (SRB, Sect. 3.1), which shapes the backbone, and Logit Pre-Sampling (LPS, Sect. 3.3), which holds down the classifier.

3.1 Spatial Redistribution Bottleneck (SRB)

A convolutional network's parameters grow with its channels, and most of that growth lands in the pointwise convolutions that mix information across channels. The inverted residual bottleneck (IRB) [29] works well on mobile devices but, leaning on pointwise convolutions and channel expansion, becomes parameter-heavy in the wide later stages. The SRB attacks this by shifting part of the channel content into the spatial domain before the bottleneck (Fig. 1a): a subset of channels is deterministically moved into a larger spatial grid, so the pointwise convolutions that follow see fewer channels, with nothing compressed or discarded.

Formally, let $X \in \mathbb{R}^{H\times W\times C}$ be partitioned along the channel dimension into $N = C/G$ groups $X = [X_1, \ldots, X_N]$, with each $X_i \in \mathbb{R}^{H\times W\times G}$. We define a deterministic operator f acting on tensor indices, inducing a bijection between the

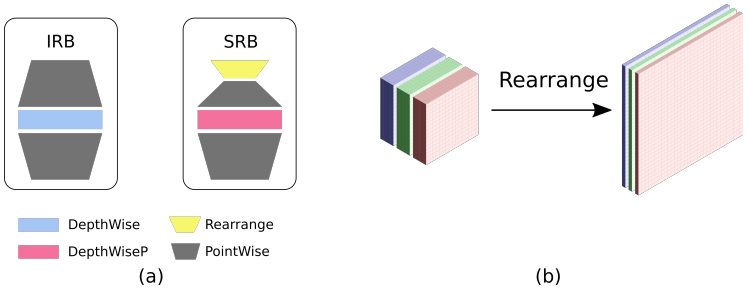

Fig. 1: Architecture and channel-to-spatial redistribution. (a) IRB versus SRB: the SRB adds the redistribution operation and a depthwise pooling layer (DepthWiseP) to keep spatial alignment for the skip connection. (b) The redistribution systematically moves channel-wise features into spatial locations.

index sets $\{1,\dots,H\}\times\{1,\dots,W\}\times\{1,\dots,G\}$ and $\{1,\dots,s_hH\}\times\{1,\dots,s_wW\}\times\{1,\dots,G/(s_hs_w)\}$, where $s_h, s_w \in \mathbb{N}$ are spatial scaling factors. By construction the operator preserves cardinality, $HWG = (s_hH)(s_wW)\big(G/(s_hs_w)\big)$, performing a structured channel-to-spatial redistribution with no aggregation or information loss. Applying f independently to each group gives $X' = \big\|_{i=1}^{N} f(X_i) \in \mathbb{R}^{\tilde{H}\times\tilde{W}\times\tilde{C}}$, with $\tilde{H} = s_hH$, $\tilde{W} = s_wW$, and $\tilde{C} = N\,G/(s_hs_w)$.

We instantiate f with a parameter-free sub-pixel rearrangement [32], an efficient and lossless realization of the bijection defined above. The contribution is architectural: it uses channel-to-spatial redistribution as an additional scaling axis inside the residual bottleneck, a role that does not depend on the particular rearrangement chosen. With an upscale factor of 2 ($s_h = s_w = 2$) the channel count drops fourfold while each spatial side doubles. The block then runs a 1×1 pointwise convolution with a nonlinearity, a 5×5 depthwise convolution at stride 2 that reads the enlarged grid and brings the resolution back into line for the skip connection, and a final 1×1 projection.

Trading channels for spatial resolution cuts parameters and FLOPS without compressing the underlying semantics. The wider grid also gives the depthwise convolution more spatial context to work with, so it picks out finer local detail, and it nudges the feature maps into complementary roles, some tracking local structure and others holding higher-level semantics. The effect is a second scaling axis: instead of stacking depth or widening channels, an SRB network gains accuracy through denser use of the budget it has, which is what lets it push past the roughly 100K-parameter ceiling of ultra-lightweight models while staying dense and more efficient than a plain IRB (Sect. 5).

3.2 BoltNet Architecture

BoltNet consists of a convolutional stem, four progressive stages, and a classification head. Given an input of spatial size $H\times W$, the stem applies a 3×3 convolution with 32 output channels, followed by a depthwise convolution. Both

layers use stride 2 and are followed by batch normalization and HardSwish activation, reducing the feature-map resolution to $H/4 \times W/4$ before the main network stages.

The four stages contain 4, 4, 4, and 3 SRB blocks with output widths of 24, 56, 152, and 368 channels, respectively. A max-pooling layer is applied at the end of each stage, except for the last one, to progressively reduce the spatial resolution while increasing the channel capacity.

The classification head applies Logit Pre-Sampling to the final feature tensor, followed by global average pooling and a linear classifier. HardSwish [9] is used throughout the network except for the output layer, and batch normalization follows every convolutional operation.

3.3 Logit Pre-Sampling (LPS)

In traditional image classification architectures, the final linear classifier can account for a substantial fraction of the total parameter budget, and this imbalance grows more pronounced as the number of classes increases. Given a pre-logit representation with C channels and a classification problem with N_{cls} classes, a conventional linear head contains CN_{cls} weights. On Pl@ntNet-300K, where $N_{\text{cls}} = 1{,}081$, the classifier can therefore become disproportionately large relative to an ultra-lightweight backbone.

Logit Pre-Sampling (LPS) reduces this cost by applying a parameter-free channel-to-spatial rearrangement to the final feature tensor before global average pooling. Let $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$ denote the pre-logit feature map, and let r be the LPS sampling factor, with C divisible by r^2. LPS applies a rearrangement operator $\mathcal{P}_r$ [32],

$$\mathbf{X}' = \mathcal{P}_r(\mathbf{X}), \qquad \mathbf{X}' \in \mathbb{R}^{rH \times rW \times C/r^2},$$

which is bijective and discards no feature values, changing only their organization across the channel and spatial dimensions.

Global average pooling is then applied to the rearranged tensor,

$$\mathbf{z} = \mathrm{GAP}(\mathbf{X}') \in \mathbb{R}^{C/r^2},$$

and the classifier is

$$\mathbf{y} = \mathbf{W}\mathbf{z} + \mathbf{b}, \qquad \mathbf{W} \in \mathbb{R}^{N_{\text{cls}} \times C/r^2}.$$

Ignoring the bias term, LPS reduces the classifier parameter count from CN_{cls} to CN_{cls}/r^2. The reduction therefore grows with both the sampling factor and the number of output classes, making LPS particularly relevant to high-cardinality recognition tasks.

The rearrangement itself is lossless, while the subsequent global pooling performs a structured aggregation of groups of pre-logit channels. LPS does not alter the convolutional backbone and introduces no learnable parameters. Its effect on model size and predictive performance is evaluated independently in Sect. 5.

4 Experiments

4.1 Benchmark Datasets

The evaluation spans three complementary environmental recognition settings. Pl@ntNet-300K is the primary benchmark and targets species-level identification from citizen-science imagery. AIDERv2 and CLRS provide secondary evaluations on aerial disaster recognition and remote-sensing scene classification, respectively, allowing us to assess whether the proposed architecture remains effective across different image domains, label-space sizes, and acquisition conditions.

Pl@ntNet-300K [6]. Pl@ntNet-300K is derived from observations submitted to the Pl@ntNet citizen-science platform and contains 306,146 images covering 1,081 plant species from 303 genera. The images are collected under uncontrolled field conditions and vary considerably in viewpoint, background, scale, composition, and visible plant organs. The dataset also exhibits substantial label ambiguity and a strongly long-tailed distribution, with the least represented 80% of species accounting for only 11% of the images. These characteristics make it a demanding benchmark for fine-grained species recognition and particularly suitable for evaluating whether an ultra-lightweight model can retain sufficient discriminative capacity over a large taxonomic label space. We use the official split of 243,916 training, 31,118 validation, and 31,112 test images.

AIDERv2 [33]. AIDERv2 is an aerial-image benchmark developed for disaster recognition and emergency monitoring. It contains approximately 16,000 multi-resolution images assigned to four classes: collapsed buildings, fires, floods, and normal scenes. The images originate from heterogeneous aerial sources and cover a range of environments and acquisition conditions, including variations in viewpoint, scene scale, resolution, and background content. From an environmental perspective, the dataset represents event-level recognition, where a model must distinguish hazardous situations from normal conditions in imagery that may be acquired by UAVs during rapid situational assessment. We use the standard 80%/10%/10% training, validation, and test split.

CLRS [14]. CLRS is a remote-sensing scene-classification dataset containing 15,000 RGB images uniformly distributed across 25 categories, with 600 images per class. Each image has a spatial resolution of 256×256 pixels, while the metric resolution of the images ranges from $0.26m$ to $8.85m$. Its categories cover a broad set of land-cover and land-use patterns, including natural, agricultural, urban, and infrastructure environments. CLRS therefore complements the object-level and event-level settings of Pl@ntNet-300K and AIDERv2 with landscape-level visual interpretation from overhead imagery. We use the dataset as a balanced secondary benchmark and adopt a 70%/10%/20% training, validation, and test split.

4.2 Training Settings

All images are normalized and resized to 224×224 pixels. For Pl@ntNet-300K, we apply a restrained combination of geometric, noise, and photometric transformations to preserve fine-grained visual characteristics. For AIDERv2 and CLRS, we follow the augmentation protocol introduced by Kyrkou *et al.* [13].

BoltNet is trained from scratch for 300 epochs using SGD with momentum 0.9 and a batch size of 256. We minimize cross-entropy loss with label smoothing of 0.1 and weight decay of 5×10^{-5}. The learning rate follows a cosine-annealing schedule from an initial value of $\eta_{\max} = 0.05$ to $\eta_{\min} = 8 \times 10^{-5}$. All experiments use random seed 22.

For the model comparison, every architecture is trained from randomly initialized weights on the target dataset. Each baseline retains the architecture-specific optimization recipe reported in its original publication, while the dataset split, input resolution, augmentation pipeline, and random seed are held fixed across models.

4.3 Evaluation Protocol and Hardware

We evaluate predictive performance using the weighted F_1 score, which is more informative than accuracy in the presence of class imbalance. Model complexity is reported in terms of parameter count and FLOPS. However, since these theoretical metrics do not reliably predict deployment cost [19], our analysis focuses primarily on measurements collected on the target hardware. For each model–platform pair, we report mean inference throughput in frames per second (FPS), average power consumption in watts (W), and energy efficiency as FPS/W.

The evaluated architectures cover three model families: mobile CNNs, including MobileNet, EfficientNet, and RegNet variants; ultra-lightweight edge models, including EmergencyNet and TakuNet; and convolution–attention hybrids, including MobileViT and EfficientFormer variants. Mobile and ultra-lightweight architectures are analyzed separately because the latter satisfy substantially tighter complexity constraints, with model sizes below 2 MB and computational costs below 100 M FLOPS.

Hardware. To assess whether model efficiency is preserved across heterogeneous deployment backends, we benchmark all compatible models on three representative edge platforms: a Raspberry Pi 5 with an ARM Cortex-A76 CPU, an NVIDIA Jetson Orin Nano 8G with an Ampere GPU, and a Hailo-8 dataflow NPU. These platforms represent general-purpose CPU execution, GPU-based parallel acceleration, and dedicated neural-network acceleration, respectively. Evaluating the same model families across these targets provides a more realistic assessment of deployment efficiency than hardware-independent complexity metrics alone.

4.4 Accuracy-Compression Tradeoff (ACT)

As a complementary read on how gracefully a model trades accuracy for size, we report the Accuracy-Compression Tradeoff (ACT), a dimensionless score that

penalizes accuracy loss non-linearly and rewards parameter reduction with diminishing returns:

$$ACT = \left(\frac{\mathrm{acc}_{\mathrm{comp}}}{\mathrm{acc}_{\mathrm{orig}}}\right)^{k} \cdot \log_2\left(1 + \frac{p_{\mathrm{orig}}}{p_{\mathrm{comp}}}\right), \tag{1}$$

where acc is the metric and p the parameter count, with subscripts for the original and compressed models. The fidelity term $(\mathrm{acc}_{\mathrm{comp}}/\mathrm{acc}_{\mathrm{orig}})^k$ makes accuracy loss bite geometrically, the exponent k setting how strongly fidelity outweighs size; the efficiency term $\log_2(1+p_{\mathrm{orig}}/p_{\mathrm{comp}})$ measures the gain from compression, and the unit shift keeps it non-negative. We interpret ACT only as a diagnostic metric. It is designed exclusively to identify, within the same model family, the architecture that most effectively compresses predictive performance into its parameter budget. Consequently, we do not use it as a general criterion for comparing different architectures or selecting the best overall model, but rather as a tool to better understand the effects of architectural design choices, alongside F_1 and the on-device measurements.

5 Results

5.1 On-device efficiency

Table 1 reports end-to-end inference on the three boards, and the ranking shifts with the execution model. BoltNet takes the best energy efficiency on the Jetson GPU (30.7 FPS/W) and the Hailo NPU (2354 FPS/W), and is a close second on the Raspberry Pi CPU (8.9 against TakuNet's 9.2). Each competitor marks a different limit. TakuNet wins the CPU, where parallelism is scarce, but its sparse blocks underuse wide accelerators, so its NPU and GPU efficiency fall to roughly 0.4 and 0.7 times BoltNet's at comparable model size. RegNet, a wide dense backbone, posts the highest raw NPU frame rate (5795 FPS) because regular convolutions map cleanly onto the dataflow array, yet it sustains this at 2.5 W against BoltNet's 1.6 W, so the two reach almost the same FPS/W with BoltNet drawing about a third less power. EmergencyNet falls in between: its atrous depthwise fusion enlarges the receptive field cheaply but scatters memory access, keeping its efficiency below BoltNet on every board. MobileViT V2 is the slowest and least efficient model everywhere, about 19× below BoltNet on the NPU, because attention's matrix products and reshapes still lack efficient edge kernels. BoltNet is the only network that stays at or near the efficiency frontier on all three execution models, a desirable aspect for a single deployable backbone.

These differences follow arithmetic intensity rather than nominal cost. The SRB narrows channels through a lossless rearrangement instead of grouped, atrous, or sparse operators, so the work that remains is standard dense pointwise and depthwise convolution, which keeps the accelerators busy. Designs that save FLOPS through irregular operators instead issue many small, memory-bound kernels that stall on parallel hardware, which is why TakuNet leads the scalar

Model	Raspberry Pi 5 *(CPU - Cortex A76)*			Hailo 8 *(NPU - Hailo Arch)*			Jetson Orin Nano *(GPU - Tegra Ampere)*		
	FPS	**W**	**FPS/W**	**FPS**	**W**	**FPS/W**	**FPS**	**W**	**FPS/W**
MobileVitV2	27.9	9.2	3.0	130.3	1.1	123.4	148.0	11.7	12.7
RegNet	72.2	9.3	7.7	**5794.7**	2.5	<u>2317.9</u>	<u>302.6</u>	12.0	<u>25.2</u>
EmergencyNet	79.8	9.1	8.8	1698.1	1.8	962.6	284.3	11.9	23.9
TakuNet	**104.6**	11.4	**9.2**	1440.4	1.5	988.6	252.4	11.4	22.1
BoltNet	<u>94.0</u>	10.6	<u>8.9</u>	<u>3778.8</u>	1.6	**2354.4**	**325.4**	10.6	**30.7**

Table 1: Inference benchmarks on three edge platforms of models trained on Pl@ntNet300K, with **best** and <u>second best</u> highlighted.

CPU but neither the GPU nor the NPU. That a model of TakuNet's size can be 2.4× less efficient than BoltNet on the NPU is the clearest single sign that parameter count and FLOPS do not predict deployed cost, and that the on-device numbers are the ones to trust.

5.2 Accuracy on long-tailed species recognition

On Pl@ntNet-300K (Table 2) BoltNet reaches 0.682 F1-score at 0.34 M parameters, the best of the deployable models below 2 MB: ahead of EmergencyNet (0.636) with 8% fewer parameters and half the FLOPS, and far ahead of TakuNet (0.483). Two design choices account for the margin over the other ultra-lightweight models. The SRB relocates capacity into the spatial grid rather than removing it, so a small budget stays usable, and Logit Pre-Sampling keeps the head from consuming that budget: on 1,081 classes a plain classifier over the final 368 channels would need about 0.40 M weights, more than the whole network, and LPS cuts this roughly fourfold. The ablation makes the effect concrete, with the head accounting for the 0.30 M-parameter gap between the SRB-only model and BoltNet. The earlier ultra-lightweight networks have neither mechanism, so their already small backbones are further starved by an oversized head. Against larger networks BoltNet is competitive rather than dominant: it trails RegNet (0.702) by two F1-score points at one-eighth the parameters, slightly exceeds FBNetV3 (0.678) at one twenty-fifth of its size, and is above MobileNetV2, MobileNetV3, EfficientNet-B0, and EfficientNetV2-S, several of which do not turn their nominal capacity into accuracy when trained from scratch on this long tail dataset (Sect. 4.2). The attention models are the most accurate, MobileViT at 0.741 and 0.732 and EfficientFormer at 0.714, since a global receptive field aids fine-grained discrimination, but Table 1 places them last for deployment by a wide margin. The benchmark is what gives the result weight: its label ambiguity and steep long tail penalize low-capacity models [6], which is why the prior ultra-lightweight networks fall so far short of BoltNet here.

Model	Parameters	Model Size (MB)	F_1-score	FLOPS (G)
EfficientNetV2 S [36]	21,562,249	85.63	0.443	2.874
FBNetV3 [5]	8,759,249	34.85	0.678	0.422
MobileNetV3 L-100 [9]	5,586,793	22.25	0.434	0.226
EfficientNet B0 [35]	5,392,309	21.40	0.493	0.399
MobileNetV2 100 [29]	3,608,633	14.30	0.444	0.314
EfficientFormerV2 S [15]	3,628,930	14.42	0.714	0.407
RegNetX 002 [23]	2,714,681	10.78	0.702	**0.203**
MobileVitV2 050 [22]	1,391,410	5.53	**0.741**	0.374
MobileVit XXS [21]	**1,298,025**	**5.18**	0.732	0.263
EmergencyNet [13]	369,647	1.48	0.636	0.116
TakuNet [25]	**297,001**	**1.19**	0.483	**0.032**
BoltNet	341,254	1.37	**0.682**	0.056

Table 2: Pl@ntNet-300K [6]: **best** and second-best per group.

Model	Parameters	Model Size (MB)	F_1-score	FLOPS (G)
EfficientNetV2 s [36]	20,182,612	80.11	0.817	2.873
FBNetV3 [5]	6,621,404	26.30	0.924	0.419
MobileNetV3 L-100 [9]	4,207,156	16.73	0.961	0.224
EfficientNet-B0 [35]	4,012,672	15.88	0.949	0.398
EfficientFormerV2 S [15]	3,247,672	12.90	**0.968**	0.407
RegNetX 002 [23]	2,317,268	9.19	0.946	**0.203**
MobileNetV2 [29]	2,228,996	8.78	0.956	0.313
MobileViT V2 050 [22]	1,114,621	4.43	0.966	0.374
MobileVit XXS [21]	**952,308**	**3.79**	0.962	0.263
EmergencyNet [13]	90,704	0.36	0.952	0.062
TakuNet [25]	**37,444**	**0.15**	0.953	**0.031**
BoltNet	241,093	0.96	**0.958**	0.055

Table 3: AIDERv2 [33]: **best** and second-best per group.

5.3 Cross-domain generalization under scarce data

Tables 3 and 4 are transfer checks, not a second contribution: they ask whether a design tuned for plant recognition still behaves well under domain shift and on the smaller environmental datasets common at the edge. On AIDERv2 the few classes and limited data saturate most models within a narrow band, and inside it BoltNet is the best deployable convolutional network (0.958 F1); the enlarged spatial grid from the SRB plausibly helps with the small local structures typical of aerial views. The harder, higher-cardinality CLRS is more discriminating: BoltNet reaches 0.825, on par with MobileViT V2 (0.826) at about one-fifth of the parameters and one-seventh of the FLOPS, and ahead of every ultra-lightweight rival. The behavior measured on Pl@ntNet-300K therefore carries

Model	Parameters	Model Size (MB)	F_1-score	FLOPS (G)
EfficientNetV2 S [36]	20,209,513	80.22	0.707	2.873
FBNetV3 [5]	6,663,089	26.47	0.668	0.420
MobileNetV3 L-100 [9]	4,234,057	16.84	0.768	0.224
EfficientNet B0 [35]	4,039,573	15.99	0.780	0.398
EfficientFormerV2 S [15]	3,255,106	12.93	0.833	0.407
RegNetX 002 [23]	2,325,017	9.22	0.836	**0.203**
MobileNetV2 100 [29]	2,255,897	8.89	0.603	0.313
MobileVitV2 050 [22]	1,120,018	4.45	0.826	0.374
MobileVit XXS [21]	**959,049**	**3.82**	**0.860**	0.264
EmergencyNet [13]	96,143	0.38	0.773	0.063
TakuNet [25]	**42,505**	**0.17**	0.760	**0.031**
BoltNet	243,046	0.97	**0.825**	0.055

Table 4: CLRS [14]: **best** and second-best per group.

over to aerial and remote-sensing imagery without modification, which is the evidence we ask these datasets to provide.

	SRB	k_{SRB}	LPS	k_{LPS}	Parameters	FLOPS	F_1-score	ACT
IRBNet					1,486,029	154.4M	0.704	1.000
	✓	2			639,610 −56.9%	56.1M −63.7%	0.687 −2.4%	1.460
	✓	3*			541,930 −63.5%	40.8M −73.6%	0.628 −10.8%	0.856
			✓	2	1,187,673 −20.1%	154.1M −0.2%	0.715 +1.56%	1.305
			✓	3*	1,136,183 −23.5%	154.2M −0.1%	0.721 +2.41%	1.426
BoltNet	✓	2	✓	2	341,254 −77.0%	55.8M −63.9%	0.682 −3.1%	1.938

Table 5: Ablation of BoltNet on Pl@ntNet-300K. SRB = Spatial Redistribution Bottleneck, LPS = Logit Pre-Sampling, k = block upscale/sampling factor. IRBNet is the inverted-bottleneck-only baseline.

5.4 Ablation and compression diagnostic

Table 5 separates the components on Pl@ntNet-300K against an inverted-bottleneck baseline (IRBNet). The SRB carries the backbone compression: at upscale factor 2 it removes 56.9% of the parameters and 63.7% of the FLOPS for a 2.4% relative drop in F1, whereas factor 3 costs 10.8%, so the redistribution has a capacity floor and must be sized to the task. LPS behaves differently depending on where it acts. On the full backbone it slightly improves accuracy (+1.6% at factor 2) while removing a fifth of the parameters, because shrinking an oversized head also regularizes it. Placed on top of the already-compressed SRB backbone, that

accuracy gain no longer transfers: the step from the SRB-only model to BoltNet removes the 0.30 M-parameter head for a 0.005 drop in F1 (0.687 to 0.682). The two components are thus complementary in what they compress, the backbone and the head, rather than additive in accuracy, and together they reach -77% parameters at -3.1% F1. The ACT curve (Fig. 2), with sensitivity $k = 7$ fixed against established backbones such as ResNet [8], ranks BoltNet highest, consistent with a design that relocates capacity rather than discarding it. Indeed, as the sensitivity increases, accuracy retention gets stricter over a fixed compression ratio, highlighting models which effectively retain accuracy despite the reduced internal representation capacity. Entries marked ($\star$) adjust widths to the operator's channel-divisibility requirement, a multiple of the squared upscale factor.

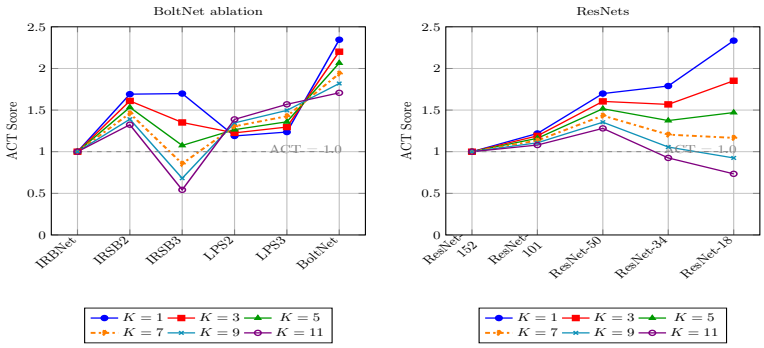

Fig. 2: Accuracy-Compression Tradeoff (ACT, ↑) for BoltNet ablations and the ResNet family. As the sensitivity coefficient k grows, the models with a superior accuracy-parameter tradeoff emerge.

6 Conclusions

We presented BoltNet, an ultra-lightweight architecture for plant species identification on constrained field hardware. Its Spatial Redistribution Bottleneck moves channel content into the spatial grid through a parameter-free, lossless rearrangement, cutting parameters and computation while letting the depthwise convolutions resolve finer local detail, and Logit Pre-Sampling applies the same idea to keep the classifier head small. On the fine-grained, long-tailed Pl@ntNet-300K benchmark BoltNet is the most accurate model in its deployable size class, and across a CPU, a GPU, and an NPU it delivers the best measured energy efficiency on the GPU and the NPU and near-best on the CPU; transfer checks on AIDERv2 and CLRS show the design holds up under domain shift and with scarce data. Rather than claiming an ecological outcome, we offer BoltNet as an efficient, deployable backbone on which field identification and, in time, on-device phenotyping systems can be built.

References

1. Boho, D., Rzanny, M., Wäldchen, J., Nitsche, F., Deggelmann, A., Wittich, H.C., Seeland, M., Mäder, P.: Flora capture: a citizen science application for collecting structured plant observations. BMC bioinformatics **21**(1), 576 (2020)
2. Burrello, A., Garofalo, A., Bruschi, N., Tagliavini, G., Rossi, D., Conti, F.: Dory: Automatic end-to-end deployment of real-world dnns on low-cost iot mcus. IEEE Transactions on Computers **70**(8), 1253–1268 (2021)
3. Capotondi, A., Rusci, M., Fariselli, M., Benini, L.: Cmix-nn: Mixed low-precision cnn library for memory-constrained edge devices. IEEE Transactions on Circuits and Systems II: Express Briefs **67**(5), 871–875 (2020)
4. Chandra, A.L., Desai, S.V., Guo, W., Balasubramanian, V.N.: Computer vision with deep learning for plant phenotyping in agriculture: A survey. arXiv preprint arXiv:2006.11391 (2020)
5. Dai, X., Wan, A., Zhang, P., Wu, B., He, Z., Wei, Z., Chen, K., Tian, Y., Yu, M., Vajda, P., et al.: Fbnetv3: Joint architecture-recipe search using predictor pretraining. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16276–16285 (2021)
6. Garcin, C., Joly, A., Bonnet, P., Lombardo, J.C., Affouard, A., Chouet, M., Servajean, M., Lorieul, T., Salmon, J.: Pl@ntnet-300k: a plant image dataset with high label ambiguity and a long-tailed distribution. In: NeurIPS 2021-35th Conference on Neural Information Processing Systems (2021)
7. Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., Xu, C.: Ghostnet: More features from cheap operations. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1580–1589 (2020)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
9. Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al.: Searching for mobilenetv3. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1314–1324 (2019)
10. Howard, A.G.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
11. Joly, A., Goëau, H., Bonnet, P., Bakić, V., Barbe, J., Selmi, S., Yahiaoui, I., Carré, J., Mouysset, E., Molino, J.F., et al.: Interactive plant identification based on social image data. Ecological Informatics **23**, 22–34 (2014)
12. Katal, N., Rzanny, M., Mäder, P., Wäldchen, J.: Deep learning in plant phenological research: A systematic literature review. Frontiers in Plant Science **13**, 805738 (2022)
13. Kyrkou, C., Theocharides, T.: Emergencynet: Efficient aerial image classification for drone-based emergency monitoring using atrous convolutional feature fusion. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **13**, 1687–1699 (2020)
14. Li, H., Jiang, H., Gu, X., Peng, J., Li, W., Hong, L., Tao, C.: Clrs: Continual learning benchmark for remote sensing image scene classification. Sensors **20**(4), 1226 (2020)
15. Li, Y., Hu, J., Wen, Y., Evangelidis, G., Salahi, K., Wang, Y., Tulyakov, S., Ren, J.: Rethinking vision transformers for mobilenet size and speed. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 16889–16900 (2023)

16. Li, Y., Yuan, G., Wen, Y., Hu, J., Evangelidis, G., Tulyakov, S., Wang, Y., Ren, J.: Efficientformer: Vision transformers at mobilenet speed. Advances in Neural Information Processing Systems **35**, 12934–12949 (2022)
17. Lin, J., Chen, W.M., Cai, H., Gan, C., Han, S.: Memory-efficient patch-based inference for tiny deep learning. Advances in Neural Information Processing Systems **34**, 2346–2358 (2021)
18. Lusk, D., Wolf, S., Svidzinska, D., Dormann, C.F., Kattge, J., Bruelheide, H., Sabatini, F.M., Damasceno, G., Moreno Martínez, Á., Violle, C., et al.: Crowd-sourced biodiversity monitoring fills gaps in global plant trait mapping. Nature communications **17**(1), 1203 (2026)
19. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: Proceedings of the European conference on computer vision (ECCV). pp. 116–131 (2018)
20. Mastin, A.J., Gottwald, T.R., van Den Bosch, F., Cunniffe, N.J., Parnell, S.: Optimising risk-based surveillance for early detection of invasive plant pathogens. PLoS biology **18**(10), e3000863 (2020)
21. Mehta, S., Rastegari, M.: Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. arXiv preprint arXiv:2110.02178 (2021)
22. Mehta, S., Rastegari, M.: Separable self-attention for mobile vision transformers. arXiv preprint arXiv:2206.02680 (2022)
23. Radosavovic, I., Kosaraju, R.P., Girshick, R., He, K., Dollár, P.: Designing network design spaces. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10428–10436 (2020)
24. Remelgado, R., Beckmann, M., Vítězslav, M., Padulosi, E., Perrone, M., Vihervaara, P., Marrs, C., Eltner, A., Rocchini, D., Cord, A.F.: Narrowing farmland biodiversity knowledge gaps with digital agriculture. npj Sustainable Agriculture **4**(1), 10 (2026)
25. Rossi, D., Borghi, G., Vezzani, R.: Takunet: an energy-efficient cnn for real-time inference on embedded uav systems in emergency response scenarios. In: 2025 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW). pp. 339–348. IEEE (2025)
26. Rusci, M., Capotondi, A., Benini, L.: Memory-driven mixed low precision quantization for enabling deep network inference on microcontrollers. Proceedings of Machine Learning and Systems **2**, 326–335 (2020)
27. Rzanny, M., Bebber, A., Wittich, H.C., Fritz, A., Boho, D., Mäder, P., Wäldchen, J.: More than rapid identification—free plant identification apps can also be highly accurate. People and Nature **6**(6), 2178–2181 (2024)
28. Rzanny, M., Mäder, P., Deggelmann, A., Chen, M., Wäldchen, J.: Flowers, leaves or both? how to obtain suitable images for automated plant identification. Plant methods **15**(1), 77 (2019)
29. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510–4520 (2018)
30. Seeland, M., Rzanny, M., Alaqraa, N., Wäldchen, J., Mäder, P.: Plant species classification using flower images—a comparative study of local feature representations. PloS one **12**(2), e0170629 (2017)
31. Seeland, M., Rzanny, M., Boho, D., Wäldchen, J., Mäder, P.: Image-based classification of plant genus and family for trained and untrained plant species. BMC bioinformatics **20**(1), 4 (2019)

32. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1874–1883 (2016)
33. Shianios, D., Kyrkou, C., Kolios, P.S.: A benchmark and investigation of deep-learning-based techniques for detecting natural disasters in aerial images. In: International Conference on Computer Analysis of Images and Patterns. pp. 244–254. Springer (2023)
34. Sze, V., Chen, Y.H., Yang, T.J., Emer, J.S.: Efficient processing of deep neural networks: A tutorial and survey. Proceedings of the IEEE **105**(12), 2295–2329 (2017)
35. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
36. Tan, M., Le, Q.: Efficientnetv2: Smaller models and faster training. In: International conference on machine learning. pp. 10096–10106. PMLR (2021)
37. Wäldchen, J., Mäder, P.: Machine learning for image based species identification. Methods in Ecology and Evolution **9**(11), 2216–2225 (2018)
38. Wäldchen, J., Mäder, P.: Plant species identification using computer vision techniques: A systematic literature review. Archives of computational methods in engineering **25**(2), 507–543 (2018)
39. Yang, Z., He, W., Fan, X., Tjahjadi, T.: Plantnet: transfer learning-based fine-grained network for high-throughput plants recognition. Soft Computing **26**(20), 10581–10590 (2022)