Proceedings of CONF-SEML 2026 Symposium: Computational Analysis and Modeling in Complex Intelligent Systems
DOI: 10.54254/2755-2721/2026.GU33477

The MobileNet Family: Principles of Lightweight Design and Edge Deployment

Pengyu Chen

Department of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China
h339732511qwe@bupt.edu.cn

Abstract. Convolutional Neural Networks (CNNs) have reshaped image recognition, but many successful CNN models come with a cost: high parameter counts and heavy computation. For real-world edge deployment—drones, phones, or medical sensors—that cost often becomes unacceptable. MobileNet emerged as an alternative, pivoting the focus from raw power to structural economy. This paper reviews the MobileNet family, focusing on three key bottlenecks (V2), and the micro-structural refinements in V3. Its practical application in agricultural pathology and clinical decision support has been rigorously validated through diverse implementations, proving its stable operation on limited Random Access Memory (RAM) (e.g., the Raspberry Pi or the older smartphones) and superior performance in resource-scarce environments. The findings show that the MobileNet can narrow the gap between realistic deployment and cutting-edge AI. This analysis substantiates that the MobileNet architecture serves as a low-cost, high-precision backbone for the next generation of the edge Artificial Intelligence (AI).

Keywords: MobileNet, depthwise separable convolution, smart agriculture, digital healthcare

1. Introduction

Computer vision has seen a surge of interest in deep learning, largely because CNNs work well on large-scale image recognition [1]. After the initial breakthroughs, a wave of models followed: VGGNet [2], GoogLeNet [3], ResNet [4]. All these models share one characteristic: a large number of parameters and high computational demands. Their accuracy is high, but that comes with a significant computational cost. For latency-sensitive applications like robotics or autonomous driving, quick response is essential. In those cases, large models often struggle.

Google then proposed a different path: MobileNet [5–7]. Instead of pushing for greater depth, this series focuses on refining the model so it can run in real time without losing competitive accuracy. The core innovation in V1 is depthwise separable convolution. The idea existed earlier (Sifre's work, later extended in Xception [8,9]), but MobileNet made it systematic. A standard convolution kernel processes spatial and channel-wise information simultaneously, which is computationally heavy. Depthwise separable convolution splits that job into two lighter steps. Later in this section this paper shows the math behind this saving.

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.GU33477

Still, V1 was not perfect. The team found certain inefficiencies that limited further optimization. V2 emerged as an improvement: inverted residuals and linear bottlenecks. This sounds like a small twist on ResNet's residual learning [4], but the inversion changes the information flow in a non-obvious way. This paper will explain why inverting the residual block prevents information collapse while improving both speed and accuracy.

V3 then adds another layer of innovation. Earlier MobileNets relied heavily on human experts to tune the architecture. V3 brings in Neural Architecture Search (NAS) to automate part of that optimization. In addition, V3 introduces a new activation function and an SE (squeeze-and-excitation) module. The result is a model that is not a single clever trick and more like a Meticulously integrated system.

The rest of this paper is organized into three parts. Section 2 describes the mechanics of all three MobileNet generations and how they evolved. Section 3 examines real-world applications—especially medical and agricultural use cases—along with current challenges and future trends for edge AI. Section 4 concludes.

2. Architectural evolution and underlying principles of MobileNet

2.1. MobileNet V1

Standard convolutions mix spatial and channel-wise processing at the same time. MobileNet V1 breaks this into two separate steps: depthwise convolution first, then pointwise convolution. The result is far fewer parameters and much less computation.

Consider an input tensor of size $\mathrm{D_F} \times \mathrm{D_F} \times \mathrm{M}$

1. Depthwise stage: M independent kernels, each of size $\mathrm{D_K} \times \mathrm{D_K}$, are applied. Each kernel works on exactly one input channel—spatial filtering only, no channel mixing.

2. Pointwise stage: The output from above is passed through N kernels of size 1×1. This step mixes information across the M channels to produce N output channels.

Figure 1 illustrates this factorization

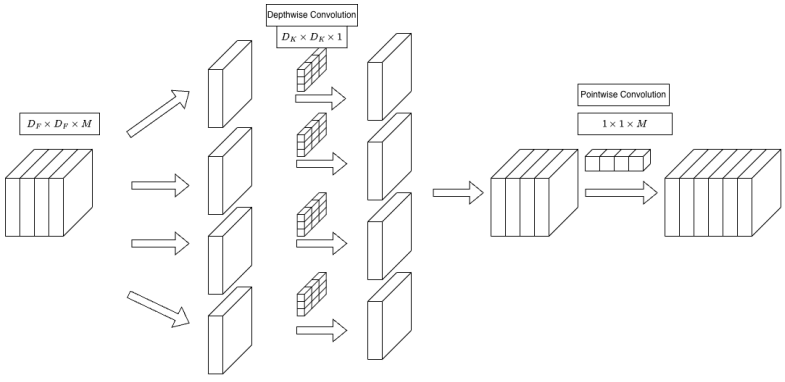


Figure 1. Mechanism of depthwise separable convolution in MobileNet V1 (picture credit: original)

Now compare the raw computation cost.A standard convolution costs: $D_F \cdot D_F \cdot M \cdot N \cdot D_K \cdot D_K$

Depthwise separable convolution splits that into two parts: $D_F \cdot D_F \cdot M \cdot D_K \cdot D_K + D_F \cdot D_F \cdot M \cdot N$

DOI: 10.54254/2755-2721/2026.GU33477

The ratio between the two is: $\frac{D_F \cdot D_F \cdot M \cdot D_K \cdot D_K + D_F \cdot D_F \cdot M \cdot N}{D_F \cdot D_F \cdot M \cdot N \cdot D_K \cdot D_K} = \frac{1}{N} + \frac{1}{D_K^2}$

As N increases, the term $1/N$ becomes negligible. For a standard 3×3 convolution ($D_K = 3$), this architecture achieves a nearly 8- to 9-fold reduction in computation.

2.2. MobileNet V2

V1 had a clear limitation: every layer is followed by a ReLU. While high-dimensional representations exhibit a degree of robustness against non-linear activations, but when ReLU is applied to low-dimensional data, information loss is inevitable—sometimes a significant amount. Researchers call this manifold collapse [6]. Non-linearity is necessary for learning complex functions, but the collapse degrades accuracy. This issue motivated V2's two main changes.

2.2.1. Inverted residuals

Standard residual blocks (like in ResNet [4]) compress first, then expand. V2 flips that.

A V2 block consists of three steps:

1. Expansion (1×1 conv, then ReLU6): expand the input to a highdimensional space. This extra room keeps information from getting crushed by ReLU.

2.Depthwise convolution (spatial filtering).

3.Projection (1×1 conv, no ReLU at the end): the tensor is compressed back to a low-dimensional manifold.

The absence of ReLU at the final projection is deliberate. After compression, the data is low-dimensional again. Adding ReLU there would discard the very information preserved by expansion. So the design leaves it out—hence the name Linear Bottleneck.

The intuition is straightforward. Important features may inherently reside on a low-dimensional manifold, but during processing they are temporarily lifted into a high-dimensional space where non-linear operations are less destructive. Then the model projects back without forcing another ReLU.

2.2.2. Skip connection

If the stride is 1 and input/output dimensions match, V2 adds a shortcut connection. The residual block becomes: $y = x + F(x)$

Taking the derivative with respect to X : $\frac{\partial y}{\partial x} = 1 + F'\left(x\right)$

Even if the weights make $F'(x)$ very small (a situation known as vanishing gradients), the constant 1 ensures that gradient signals can still propagate back to earlier layers. This is the same core idea as ResNet, applied inside an inverted structure.

2.2.3. ReLU6 for quantization

V2 also uses ReLU6 instead of plain ReLU. It clips activation values at 6 to make quantization more efficient. By bounding the range to [0,6], the model can't rely on large magnitude values to encode information. Instead, it must use fine-grained differences.

DOI: 10.54254/2755-2721/2026.GU33477

A concrete example helps illustrate the benefit.Suppose $x = 3.6$ and it is quantized to INT8 (0–255 range). The scale factor is roughly $6/255 \approx 0.023529$. The quantized integer becomes: $Q = round\left(\frac{3.6}{0.023529}\right) = 153$

During de-quantization, the integer 153 is reconstructed as 3.60009 , resulting in a negligible rounding error of 0.00009 while significantly boosting inference speed on INT8 hardware.

Figure 2 shows the V2 block structure.

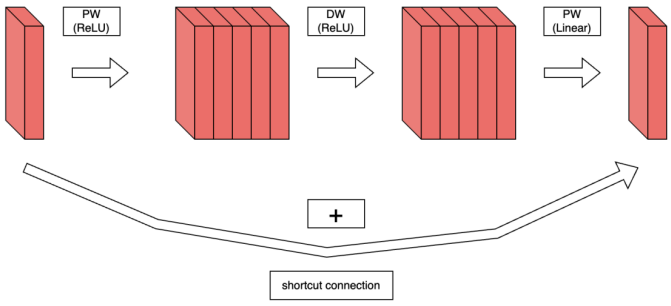


Figure 2. Mechanism of MobileNet V2 (picture credit: original)

2.3. MobileNet V3

V3 is based on what V2 built and adds three improvements: the Squeeze-and-Excitation (SE) module [10], Neural Architecture Search (NAS) [11], and a new activation function called h-swish. None of these are completely new ideas on their own, but V3 combines them and form a Highly Synergized Framework.

2.3.1. Squeeze-and-Excitation (SE)

SE is set after the depthwise convolution. Its job is simple: determine which channels matter most for the current input, then boost those and suppress the rest.

The mechanism works as follows:

1.Global average pooling squeezes the $\mathrm{D_F} \times \mathrm{D_F} \times \mathrm{M}$ feature map down to a $1 \times 1 \times \mathrm{M}$ vector. Each value in that vector summarizes one entire channel.

2.Two fully connected layers take that vector and produce a set of channelspecific weights. These weights get updated during backpropagation—so the model learns which channels to pay attention to.

3. The original feature map gets multiplied by these weights channelbychannel.

The computational cost is very small—just pooling plus two tiny linear layers. This yields adaptive feature re-weighting at almost no additional cost.

DOI: 10.54254/2755-2721/2026.GU33477

2.3.2. Neural Architecture Search (NAS)

Earlier MobileNets relied on human experts to design the network topology. Such manual processes are inherently inefficient, and manual architectural tuning often fails to identify the optimal trade-off between inference speed and predictive accuracy. V3 instead uses hardwareaware NAS—an automated search that explores a vast search space of potential topologies and identifies architectures optimized for actual mobile silicon throughput rather than mere theoretical performance.

2.3.3. Activation: from Swish to h-swish

Evidence suggests that Swish [12] can learn richer representations than ReLU. Swish was discovered through automated search, and its shape is smooth and non-monotonic. This smoothness helps gradient flow during training—unlike ReLU, which has a hard corner at zero where the derivative jumps. In addition, Swish dips slightly into negative values, so it does not discard negative feature information entirely.

But Swish is computationally expensive (it involves a sigmoid). For a mobile model, that is a problem. V3's solution is h-swish, a piecewise linear approximation of Swish that uses the already-efficient ReLU6. The formula is: $h-swish\left(x\right)=x\cdot\frac{ReLU6(x+3)}{6}$

This provides most of Swish's benefits—smooth transition, negative tail—but runs much faster because ReLU's computationally friendly.

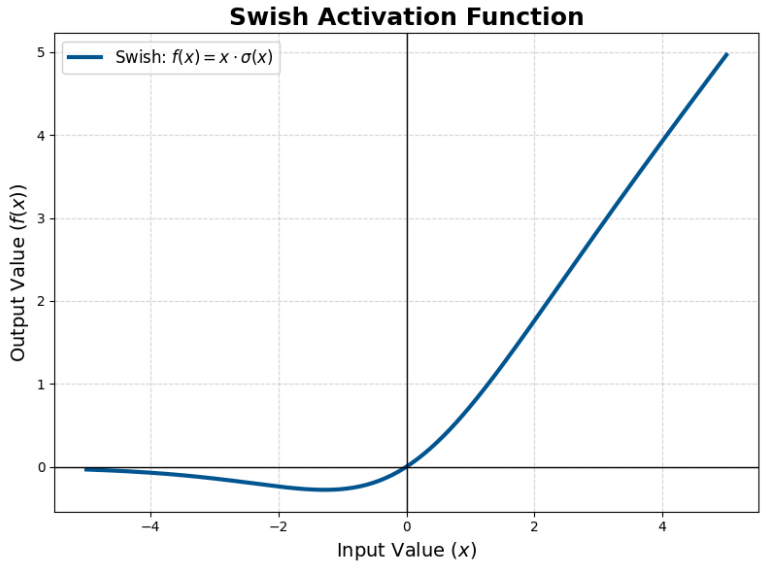


Figure 3. Swish activation function

3. Applications and discussion

3.1. Diversity of application scenarios

MobileNet's strengths—real-time speed, low computational overhead, and decent accuracy—make it a natural fit for deployment outside the data center. To validate the alignment between theoretical

DOI: 10.54254/2755-2721/2026.GU33477

efficiency and actual performance, this paper examined two very different fields: medical imaging and agricultural disease detection. Both have limited compute budgets, spotty connectivity, or both.

3.1.1. Medical imaging

Automated diagnostic systems can act as a first pass. They do not replace doctors, but they can flag high-risk cases early and reduce clinical workload. Currently, Vision Transformers (ViTs) are state of the art on many medical benchmarks. However, ViTs overfit severly when labeled data is scarce. In medicine, high-quality labeled images are precisely what is lacking.

This is where MobileNet comes in. A tailored MobileNet—with task-specific structural adjustments—can be a more practical choice.

A strong example comes from Zhu et al. in the MICCAI MMAC 2023 Challenge [13]. They used a carefully tuned MobileNet variant. The researchers adopted a bold augmentation strategy, challenging the prevailing assumption that excessive noise or distortion would inevitably compromise model stability. Their MobileNet actually beat much larger models in accuracy while remaining computationally lean. For oncology, where early detection saves lives, this kind of lightweight screening tool matters.

Indraswari et al. [14] confirmed the same pattern. They used MobileNet as a backbone with ImageNet pre-training, and the model held up under real clinical conditions. The implication is clear: efficient models like MobileNet do not just close the deployment gap—they also make top-tier AI affordable for hospitals without large GPU clusters.

3.1.2. Agricultural detection

Agricultural environments present unique challenges that differ significantly from centralized data centers, characterized by intermittent connectivity and hardware with limited computational throughput.

Historically, disease detection meant a human walking around looking at leaves—slow, exhausting, and error-prone.

Consider apple leaf disease identification [15]. Researchers chose MobileNet because it runs on actual mobile hardware without special optimizations. When compared against traditional CNNs, MobileNet achieves accuracy-versus-latency trade-off.

Some work has gone further. Chen et al. [16] added a location-wise soft attention mechanism to the V2 framework. This helps the model ignore background clutter—dirt, shadows, other leaves—and focus on actual disease signs. Combined with transfer learning, this modified MobileNet performs well even in messy real-field conditions.

3.2. Discussion

3.2.1. Limited data and overfitting

In specialized domains—medical, agricultural, industrial—rarely have millions of labeled images. Small datasets often mean models either fail to generalize or overfit severely. The standard fix is transfer learning. Pre-train on ImageNet (14 million images, 1000 classes), then fine-tune on the small target dataset. That warm start gives the model basic spatial reasoning skills, so it doesn't have to learn everything from scratch. Convergence is faster, and final accuracy is higher. Augmentation helps too: rotation, cropping, color shifts. However excessive data augmentation will cause opposite

DOI: 10.54254/2755-2721/2026.GU33477

result. For instance, performing a horizontal flip on a digit '6' yields a '9', which introduces label noise and compromises data integrity if the ground truth remains unchanged. That's not data diversity, that's data corruption. So augmentation needs to be task-aware, not just aggressive.

3.2.2. Focusing on salient feature

Most objects are defined by a handful of discriminative features. Standard CNNs tend to spread attention evenly, which means they also learn background noise. The SE module (from V3) addresses this by learning which channels are important and re-weighting them accordingly. In agriculture—where field images are often cluttered—this is a clear advantage. The model learns to focus on the lesion, not the dirt.

3.2.3. Integration with transformer

Transformers changed the field with self-attention and global receptive fields. But they often require enormous data to generalize and High-performance computing (HPC) devices to support the training and deployment. For edge deployment, pure Transformers are often impractical.

The emerging trend is hybrid architectures. FastViT is a good example [17]. It replaces standard self-attention with a RepMixer module that embeds depthwise convolutions. The result combines the best of both: Transformer-like receptive fields with CNN-like efficiency. By integrating structural reparameterization, FastViT successfully harmonizes the global receptive field of Transformers with the localized efficiency of MobileNet-style convolutions.

4. Conclusion

This paper has traced the MobileNet family from V1 to V3. Depthwise separable convolution reduced computation by nearly an order of magnitude. Inverted residuals and linear bottlenecks fixed V1's information-loss problem. V3 added SE, NAS, and h-swish—each modest on its own, but together they form a cohesive, efficient system.

This paper then validated these ideas in two real-world settings: healthcare and agriculture. In both cases, MobileNet performed well where larger models would be too slow or too heavy. Transfer learning helped with small datasets. Attention-like mechanisms (SE, soft attention) helped with noisy real-world images.

Challenges remain. Overfitting on tiny datasets is a real concern. Blind augmentation can backfire. Pure Transformers are still too heavy for most edge devices.

Nevertheless, the path forward is becoming clear: hybrid models that mix CNN efficiency with Transformer-style receptive fields. MobileNet has already demonstrated that lightweight design does not have to mean low accuracy. The next generation of edge AI will likely combine MobileNet-inspired ideas with selective attention mechanisms, running on phones or farm sensors. The main claim stands: MobileNet provides a low-cost, high-precision backbone that narrows the gap between cutting-edge AI and realistic deployment.

References

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (NIPS) (pp. 1097–1105).

[2] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv: 1409.1556.

DOI: 10.54254/2755-2721/2026.GU33477

[3] Szegedy, C., et al. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 1–9).
[4] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 770–778).
[5] Howard, A. G., et al. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv: 1704.04861.
[6] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, B. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 4510–4520).
[7] Howard, A., et al. (2019). Searching for mobilenetv3. In Proceedings of the IEEE/CVF international conference on computer vision (ICCV) (pp. 1314–1324).
[8] Sifre, L., & Mallat, S. (2014). Rigid-motion scattering for image classification (Doctoral dissertation, Ecole Polytechnique, CMAP).
[9] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 1251–1258).
[10] Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 7132–7141).
[11] Zoph, B., & Le, Q. V. (2016). Neural architecture search with reinforcement learning. In Proceedings of the international conference on learning representations (ICLR).
[12] Ramachandran, P., Zoph, B., & Le, Q. V. (2017). Searching for activation functions. arXiv preprint arXiv: 1710.05941.
[13] Zhu, W., et al. (2023). Beyond mobilenet: An improved mobilenet for retinal diseases. In International conference on medical image computing and computer-assisted intervention (MICCAI) (pp. 56–65). Springer.
[14] Indraswari, R., Rokhana, R., & Herulambang, W. (2022). Melanoma image classification based on mobilenetv2 network. Procedia Computer Science, 197, 198–207.
[15] Bi, C., et al. (2022). Mobilenet-based apple leaf diseases identification. Mobile Networks and Applications, 27(1), 172–180.
[16] Chen, J., Zhang, D., Suzauddola, M., & Zeb, A. (2021). Identifying crop diseases using attention embedded MobileNetV2 model. Applied Soft Computing, 113, 107901.
[17] Vasu, P. K. A., Gabriel, J., Zhu, J., Tuzel, O., & Ranjan, A. (2023). FastViT: A fast hybrid vision transformer using structural reparameterization. In Proceedings of the IEEE/CVF international conference on computer vision (ICCV) (pp. 5785–5795).