applied system innovation

MDPI

Article

Recursive Weight Sharing for Parameter-Efficient Deep Convolutional Networks: Application to Skin Lesion Classification

Ali Belkhiri *, My Abdelouahed Sabri and Abdellah Aarab

Faculty of Sciences Dhar-Mahraz, Sidi Mohamed Ben Abdellah University, Fez 30000, Morocco; abdelouahed.sabri@usmba.ac.ma (M.A.S.); abdellah.aarab@usmba.ac.ma (A.A.)
* Correspondence: ali.belkhiri@usmba.ac.ma

Abstract

Modern deep convolutional neural networks achieve remarkable performance but require substantial computational resources due to their large parameter counts, limiting their suitability for resource-constrained environments. We propose Tiny Recursive ResNet-50, a parameter-efficient architecture that reduces model complexity through recursive feature refinement with weight sharing across reasoning cycles. The proposed design combines lightweight bottleneck blocks, iterative latent state accumulation, and deep supervision to enhance representation quality without increasing parameter count. Extensive experiments are conducted on melanoma classification using the HAM10000 dataset as the primary training and evaluation benchmark. Results demonstrate that the proposed recursive architecture maintains competitive accuracy while reducing parameters by approximately 49%, confirming its efficiency under constrained settings. To assess robustness under limited data and acquisition variability, we additionally validate on the PH2 dataset (200 images). Due to the small dataset size and class imbalance, evaluation is performed using 5-fold stratified cross-validation, and performance metrics are reported as mean ± standard deviation. This validation confirms that recursive refinement with moderate cycle depth improves stability and generalization in small-data regimes.

Keywords: recursive neural networks; parameter efficiency; ResNet-50; deep learning architecture; iterative refinement; model compression; medical image analysis; convolutional neural networks; feature learning; computational efficiency

Academic Editors: Christos Douligeris and Teen-Hang Meen

Received: 25 February 2026
Revised: 23 March 2026
Accepted: 23 March 2026
Published: 25 March 2026

1. Introduction

The rapid advancement of deep learning in computer vision has been largely driven by increasingly complex neural network architectures. Models such as ResNet-50 [1], ResNet-152 [1], and EfficientNet [2] have demonstrated remarkable performance across a wide range of tasks; however, these gains have come at the cost of substantial computational and memory requirements. Such constraints limit their applicability in resource-constrained environments, including mobile devices, embedded systems, and edge computing platforms, where efficiency and scalability are essential.

To address these challenges, a variety of model compression techniques have been proposed, including pruning [3], quantization [4], knowledge distillation [5], and neural architecture search [6]. While these approaches can significantly reduce model size and computational cost, they typically rely on multi-stage optimization pipelines, require careful hyperparameter tuning, and may introduce performance degradation. More importantly, they treat parameter efficiency as a post hoc objective rather than as a core design principle.

An alternative perspective is offered by recursive neural networks, where model depth is achieved through the repeated application of a shared computational module rather than through stacking independent layers. This paradigm enables the progressive refinement of feature representations while maintaining a fixed parameter budget. Previous works, such as the Tiny Recursive Model (TRM) [7], have demonstrated that recursive processing can achieve competitive performance with significantly fewer parameters. Other approaches, including Feedback Networks [8] and related studies bridging residual and recurrent learning [9], further highlight the potential of iterative refinement in visual recognition tasks. Additionally, Recursive Cortical Networks [10] have explored the integration of recursive processing with probabilistic modeling, although their applicability to standard CNN backbones remains limited.

Despite these advances, the application of recursive architectures to modern convolutional backbones, such as ResNet-50, remains insufficiently explored. In particular, the relationship between recursive depth, dataset size, and generalization performance has not been systematically investigated, especially in the context of medical image analysis.

In this work, we propose Tiny Recursive ResNet-50, a parameter-efficient architecture that integrates recursive feature refinement with latent state accumulation and deep supervision. By reusing the same set of parameters across multiple refinement cycles, the model effectively increases its representational capacity without increasing its parameter count. This design enables a direct comparison with the standard ResNet-50, as the backbone structure is preserved and efficiency is introduced through temporal weight sharing. An overview of the proposed Tiny Recursive ResNet-50 architecture is illustrated in Figure 1.

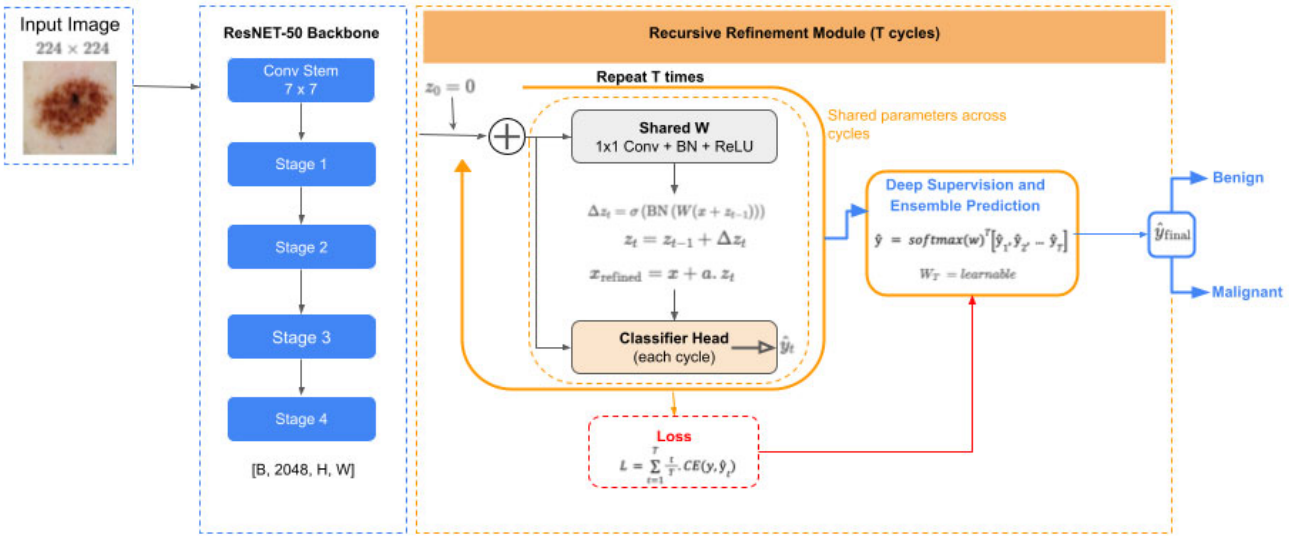

Figure 1. Architecture of the proposed Recursive ResNet-50.

We evaluate the proposed approach on the task of skin lesion classification, a clinically relevant problem where both accuracy and computational efficiency are critical. Experiments are conducted on two datasets with different characteristics: HAM10000 [11], which enables evaluation in a large-scale setting, and PH2 [12], which allows assessing generalization under limited data conditions. This dual evaluation provides a comprehensive perspective on the behavior of parameter-efficient architectures across different data regimes.

The main contributions of this work are as follows: (1) we introduce a recursive architecture that achieves approximately 49.4% parameter reduction while maintaining competitive performance; (2) we demonstrate that parameter efficiency can act as an implicit form of regularization, improving generalization and stability in small-data scenarios;

(3) we provide an extensive empirical analysis of the effect of recursive depth on performance; and (4) we highlight the practical advantages of the proposed approach for deployment in resource-constrained environments.

2. Related Work

2.1. Efficient Convolutional Neural Network Architectures

The pursuit of efficient CNN architectures has led to several research directions. MobileNets [13] introduced depthwise separable convolutions that factorize standard convolutions into depthwise and pointwise operations, thereby reducing the computational cost while maintaining accuracy. MobileNetV2 [14] adds inverted residual structures with linear bottlenecks. ShuffleNet [15] employs channel shuffle operations to enable information flow across feature channels in group convolutions. SqueezeNet [16] uses fire modules with squeeze and expand layers to achieve AlexNet-level accuracy with 50× fewer parameters.

EfficientNet [2] introduced compound scaling, which uniformly scales the network depth, width, and resolution using a principled coefficient. This approach achieved state-of-the-art accuracy with fewer parameters than those of previous models. RegNet [17] performed design space exploration to identify good network architectures and discovered that simple, regular designs often outperform complex architectures. The NFNet [18] eliminated batch normalization to improve training speed and efficiency. Although these architectures improve efficiency, they achieve this through specialized convolution operations rather than weight sharing across time, representing a fundamentally different approach from recursive architectures.

Our work differs from these approaches in that it achieves efficiency through recursive weight sharing rather than specialized convolution operations. This enables a direct comparison with standard architectures (e.g., ResNet-50) because the primary change is temporal rather than spatial processing.

2.2. Model Compression Techniques

Model compression techniques reduce the size of trained networks using post hoc modifications. Network pruning [3,19] removes redundant connections or entire neurons based on the magnitude, gradient information, or learned importance scores. Structured pruning [20] removes entire filters or layers, enabling practical speedups without the need for specialized hardware. However, pruning typically requires iterative train-prune-finetune cycles and careful sensitivity analyses to avoid accuracy degradation.

Quantization reduces numerical precision from 32-bit floating points to 8-bit integers or even binary values [4,21]. Post-training quantization [22] converts trained models with minimal accuracy loss, whereas quantization-aware training [23] simulates low-precision arithmetic during training for better accuracy. Knowledge distillation [5,24] trains a small student network to mimic the predictions or intermediate representations of a large teacher. This approach can achieve good compression ratios but requires access to the original training data and multiple training phases.

Low-rank factorization [25] decomposes weight matrices into products of smaller matrices, reducing the number of parameters while approximating the original transformation. Tensor decomposition methods [26] extend this approach to higher order tensors. Although effective, these compression techniques treat efficiency as an afterthought, requiring a fully trained model before compression. Our recursive architecture incorporates efficiency from the design phase, thereby avoiding the need for complex multi-stage training procedures.

2.3. Recursive and Recurrent Vision Models

Recursive neural networks reuse computational modules across time to construct deep representations with fewer parameters. Early recurrent convolutional networks [27] applied recurrent connections within the CNN layers for sequential processing. The Recursive Cortical Network (RCN) [10] combines recursive processing with probabilistic inference for few-shot learning. Feedback Networks [8] introduced top-down feedback connections that iteratively refine representations, showing improvements on ImageNet classification.

The Universal Transformer [7] applied recursive processing to attention-based models, demonstrating that depth can be achieved through the repeated application of the same transformation. Residual Attention Networks [28] use attention mechanisms to iteratively refine feature maps. Attention U-Net [29] applies attention gates to medical image segmentation. However, these models either target specific tasks (segmentation and sequential data) or require substantial architectural changes, complicating direct comparisons with standard baselines.

ConvRNN and ConvLSTM [30] combine convolutional operations with recurrent connections for video prediction and spatiotemporal modeling. Although effective for sequential data, these architectures target problems different from single-image classification. Existing studies on iterative refinement [31,32] has shown promise in various vision tasks, but a systematic investigation of recursive ResNet variants, particularly understanding how recursive depth interacts with dataset size and generalization, remains underexplored. Our proposed Tiny Recursive ResNet-50 fills this gap by providing a clean architectural comparison with an extensive empirical analysis across different data regimes.

3. Architecture Design

3.1. Standard ResNet-50 Baseline

ResNet-50 [1] was used as the baseline architecture for comparison. The network consists of an initial 7×7 convolutional layer with stride 2, followed by max pooling, and four residual stages containing [3, 4, 6, 3] bottleneck blocks. Each bottleneck block uses three convolutional layers (1×1, 3×3, 1×1) with expanding and contracting channel dimensions. Skip connections enable the gradient flow by adding the input directly to the block output. The architecture was concluded with a global average pooling and a fully connected classifier. The standard ResNet-50 contains 23,512,130 parameters.

The bottleneck design reduces the computational cost compared to basic residual blocks by using 1×1 convolutions to reduce the dimensionality before expensive 3×3 spatial convolutions, and then expanding back. This architecture has demonstrated excellent performance across various vision tasks and serves as a standard backbone for many applications. Its widespread adoption makes it an ideal baseline for evaluating the proposed recursive approach. Figure 2 presents the architecture of the ResNet-50.

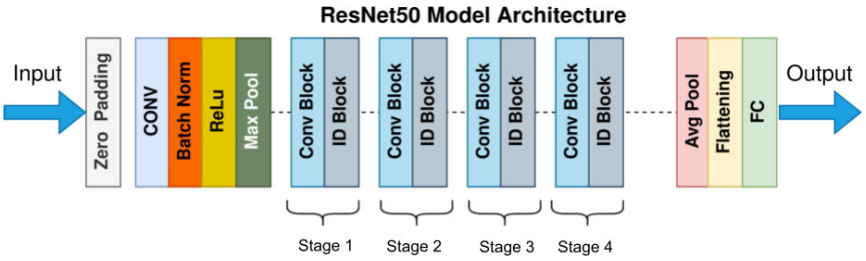

Figure 2. Architecture of the ResNet-50.

3.2. Tiny Recursive ResNet-50: Core Architecture

We propose Tiny Recursive ResNet-50, a parameter-efficient architecture that achieves depth through recursive weight sharing rather than stacking independent layers. The core idea is to iteratively refine feature representations using a shared refinement module with latent state accumulation and deep supervision, enabling substantial parameter reduction without sacrificing representational capacity. The effectiveness of this design is validated experimentally in the following sections.

- Recursive Feature Refinement with Latent State: After feature extraction through four stages, we introduce recursive processing that iteratively refines representations without additional parameters (Figure 3). Let x denote the stage 4 output feature map with shape [batch, 2048, H, W]. We initialize a latent state $z_t = 0$ of the same shape, which accumulates learned corrections across **T** recursive cycles. At each cycle t, we compute:

$$\Delta_{z_t} = \sigma(BN(W(x + z_{t-1})))$$

where W is a 1×1 convolution (2048→2048 channels), BN is the batch normalization, and σ is the ReLU activation. The latent state accumulates corrections as follows:

$$z_t = z_{t-1} + \Delta_{z_t}$$

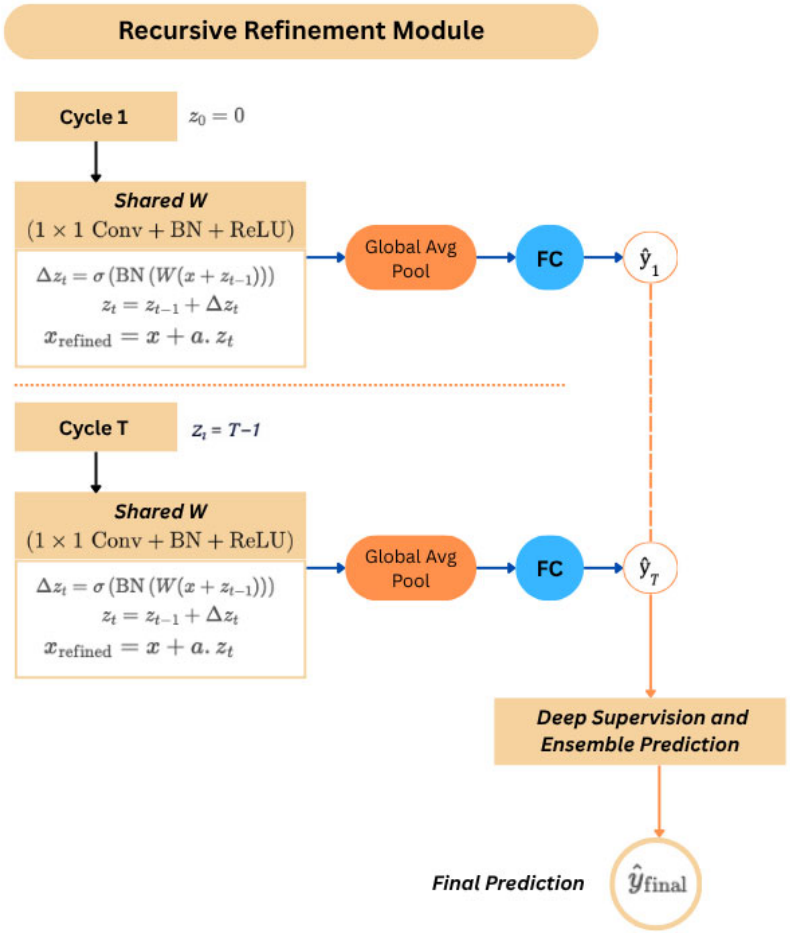

Figure 3. Recursive Refinement Module.

Refined features combine the original and accumulated corrections:

$$x_{refined} = x + \alpha \cdot z_t$$

where $\alpha = 0.3$ is a mixing coefficient. This formulation enables the network to progressively adjust its representation while maintaining a connection to the original features.

- Deep Supervision and Ensemble Prediction: Unlike standard architectures that produce a single prediction, we generated predictions at every recursive cycle through cycle-specific classification heads. During training, we computed the weighted cross-entropy loss across all cycles:

$$L = \sum_{t=1}^{T} \frac{t}{T} . CE(y, \hat{y}_t)$$

where later cycles received higher weights, encouraging progressive refinement. Each cycle has its own global average pooling and fully connected layer (2048→2 for binary classification), enabling independent predictions while sharing a recursive refinement backbone. During inference, we compute a weighted ensemble using learned softmax weights over all cycles:

$$\hat{y} = softmax(w)^T [\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_T]$$

where w is a learnable parameter. This allows the model to adaptively determine which cycles contribute most to the final predictions.

The complete architecture maintains the ResNet-50 stem (7×7 conv + max pool), four-stage structure, and identical bottleneck block configurations. The only architectural addition is the recursive refinement module applied after Stage 4. This design enables direct computational and performance comparisons because the base network remains unchanged; all parameter savings come from weight sharing across recursive cycles. The number of recursive cycles T becomes a hyperparameter that trades off the computational cost versus the refinement capacity.

The parameter reduction in Tiny Recursive ResNet-50 arises from a fundamental principle: weight sharing across time rather than across space. In a standard ResNet-50, each independent residual block owns its own weight matrices $W_1, W_2, \ldots, W_N$. The total parameter count scales linearly with the number of stages and blocks:

$$N(\theta_{standard}) = \sum_i N(W_i) \approx 23.51M$$

Therefore:

$$N(\theta_{recursive}) = N(\theta_{backbone}) + N(W_{shared}) + T \cdot N(\theta_{head})$$

where $N(\theta_{backbone}) \approx 11.4M$ (lightweight bottleneck redesign), $N(W_{shared}) \approx 0.5M$ (1×1 conv, 2048→2048), and $N(\theta_{head}) \approx 2K \times T$ (per-cycle classification heads). The total ≈11.89M is independent of T, confirming O(1) scaling with recursive depth analogous to RNNs, which process sequences of arbitrary length with fixed parameter budgets.

By contrast, the recursive refinement module applies a single shared weight matrix W_{shared} repeatedly across T cycles. The critical insight is that T iterations are processed using the same ≈0.5M parameters instead of $T \times 0.5M$ distinct ones.

This theoretical framework explains why the parameter count does not increase with more cycles: the model borrows computational depth from time rather than buying it with additional parameters. The lightweight bottleneck redesign (reduced channel widths in early stages) accounts for the reduction from 23.51M to ~11.4M in the backbone itself; the recursive module's sharing accounts for the remainder.

3.3. Experimental Validation: Skin Lesion Classification

To validate our architecture, we conducted experiments on skin lesion classification, a binary classification task (benign vs. malignant) using dermoscopic images. Medical imaging provides an excellent testbed for evaluating parameter-efficient architectures because (1) deployment constraints are strict owing to clinical environments with limited computational resources; (2) datasets span different scales, from small clinical collections to large public datasets, enabling evaluation across data regimes; and (3) task difficulty is sufficient to reveal meaningful performance differences between architectures.

The HAM10000 (Human Against Machine with 10,000 images) is used as the principal dataset for model training and large-scale evaluation. The dataset [11] contains 10,015 dermoscopic images of pigmented skin lesions collected from different populations and imaging systems. For binary classification, we grouped common and atypical nevi as benign (class 0) and melanoma as malignant (class 1), resulting in approximately 70% benign and 30% malignant cases. The dataset was split 80/20 into training (8012 images) and validation (2003 images) with stratified sampling to maintain class balance. Figure 4 presents representative dermoscopic examples from the HAM10000 dataset. This large-scale dataset tests the architecture's capacity to learn from abundant data and provides statistical significance for performance comparisons.

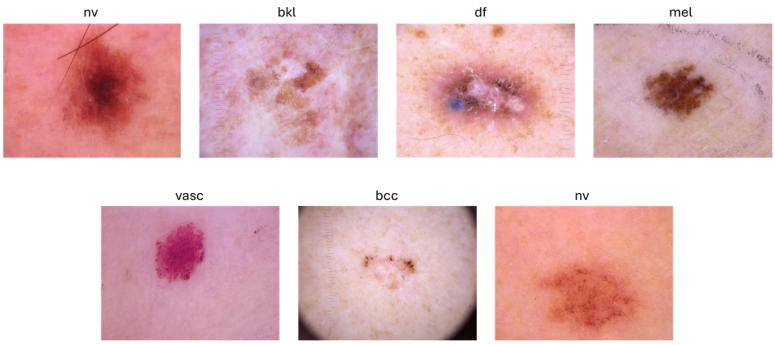

Figure 4. Representative dermoscopic examples from the HAM10000 dataset.

HAM10000 Dataset:

The PH2 dataset [12] contains 200 high-quality dermoscopic images acquired at Hospital Pedro Hispano, Portugal: 80 common nevi, 80 atypical nevi, and 40 melanoma images. Figure 5 presents representative dermoscopic examples from the PH2 dataset. After binary grouping (160 benign, 40 malignant), PH2 is evaluated exclusively using 5-fold stratified cross-validation. Given the limited dataset size and inherent class imbalance, PH2 is used exclusively for validation to evaluate model robustness under small-data conditions and distribution differences relative to HAM10000. A single hold-out split is intentionally avoided, as it can produce split-dependent and statistically unstable results.

PH2 Dataset:

Instead, we adopt 5-fold stratified cross-validation, ensuring that each fold preserves the benign/malignant class distribution. Performance metrics are reported as mean $\pm$ standard deviation across folds to quantify both accuracy and result stability.

Training Configuration: All models were trained from scratch to ensure a fair comparison of architectural efficiency. The images were resized to 224 $\times$ 224 pixels and normalized. Table 1 presents the Common training configuration used for all experiments. For HAM10000, models are trained using standard supervised learning procedures with cross-entropy loss and cosine annealing learning rate scheduling.

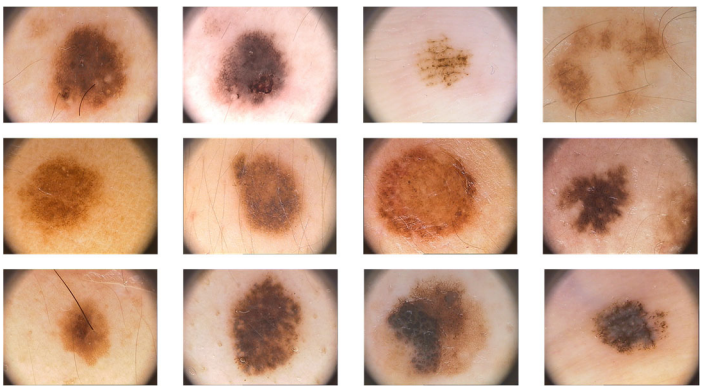

Figure 5. Representative dermoscopic examples from the Ph2 dataset.

Table 1. Common training configuration used for all experiments.

Hyperparameter	Value
Task	Binary classification (Benign vs. Malignant)
Input image size	224×224
Recursive cycles (T)	{5, 6, 7, 10}
Optimizer	Adam
Initial learning rate	0.0012
Weight decay	(1.10×10^{-4})
Batch size	32
Number of epochs	30
Learning rate schedule	Cosine annealing
Loss function	Cross-entropy
Data augmentation	Horizontal/vertical flip ($p = 0.5$), rotation ($\pm 15°$), color jitter (0.2), affine transform (0.1)
Normalization	ImageNet mean and standard deviation
Random seed	42
Hardware	Apple M2 GPU
Framework	PyTorch 2.2.2 (MPS backend)

Both the baseline ResNet-50 and the proposed Tiny Recursive ResNet-50 were trained using identical hyperparameters and training protocols to ensure a fair comparison.

For PH2, we employ stratified 5-fold cross-validation:

- The dataset is divided into five folds of equal size.
- Each fold preserves the benign/malignant proportion.
- The model is trained and evaluated five times, rotating validation folds.
- Metrics are aggregated as mean $\pm$ standard deviation.

This protocol mitigates partition sensitivity and provides statistically credible performance estimates for small datasets.

All experiments were conducted using the same training configuration to ensure a fair comparison across architectures and recursive depths. This design isolates the effect of recursion and parameter efficiency from other optimization-related factors.

Evaluation Metrics: To quantitatively evaluate the classification performance of the proposed Tiny Recursive ResNet-50 and the baseline ResNet-50, we employ standard metrics commonly used in medical image analysis: accuracy, precision, recall, F1-score,

and the area under the ROC curve (ROC-AUC). These metrics are computed based on the confusion matrix elements: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

- Accuracy measures the overall proportion of correctly classified samples:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- Precision reflects the reliability of positive predictions:

$$Precision = \frac{TP}{TP + FP}$$

- Recall measures the ability to correctly identify malignant cases:

$$Recall = \frac{TP}{TP + FN}$$

- The F1-score provides a harmonic balance between precision and recall:

$$F1 - score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

- The ROC-AUC metric evaluates the model's discrimination ability across different classification thresholds by measuring the area under the Receiver Operating Characteristic curve.

4. Experimental Results

4.1. Performance on Large-Scale Data (HAM10000)

Table 2 presents comprehensive results for HAM10000, which demonstrate that Tiny Recursive ResNet-50 achieves competitive performance with substantially fewer parameters. Both models trained for 30 epochs, with the recursive model using 5 cycles. Note that recall is listed first among the performance metrics, reflecting our position that sensitivity to malignant cases is the overriding clinical priority.

Table 2. Performance comparison using the HAM10000 dataset (10,015 images, 30 epochs).

Metric	Standard ResNet-50	Tiny Recursive (5 Cycles)	Difference
Parameters	23.51M	11.89M	−49.4%
Model Size	89.7 MB	45.4 MB	−49.4%
Accuracy	0.8805	0.8670	−1.35%
Precision	0.9359	0.8536	−8.23%
Recall (Sensitivity)	0.8170	0.8860	+6.90%
F1-Score	0.8724	0.8695	−0.29%
ROC-AUC	0.9619	0.9390	−2.29%
Training Time	18,830 s	8155 s	−56.7%

Figure 6 presents learning curves of standard ResNet-50 and recursive ResNet-50 models. The recursive architecture achieves a recall of 88.60% compared to standard ResNet-50's 81.70%, a 6.9 percentage-point improvement in malignant-case detection despite using 49.4% fewer parameters. In absolute terms, the recursive model achieves higher recall on malignant samples (88.60% vs. 81.70%).

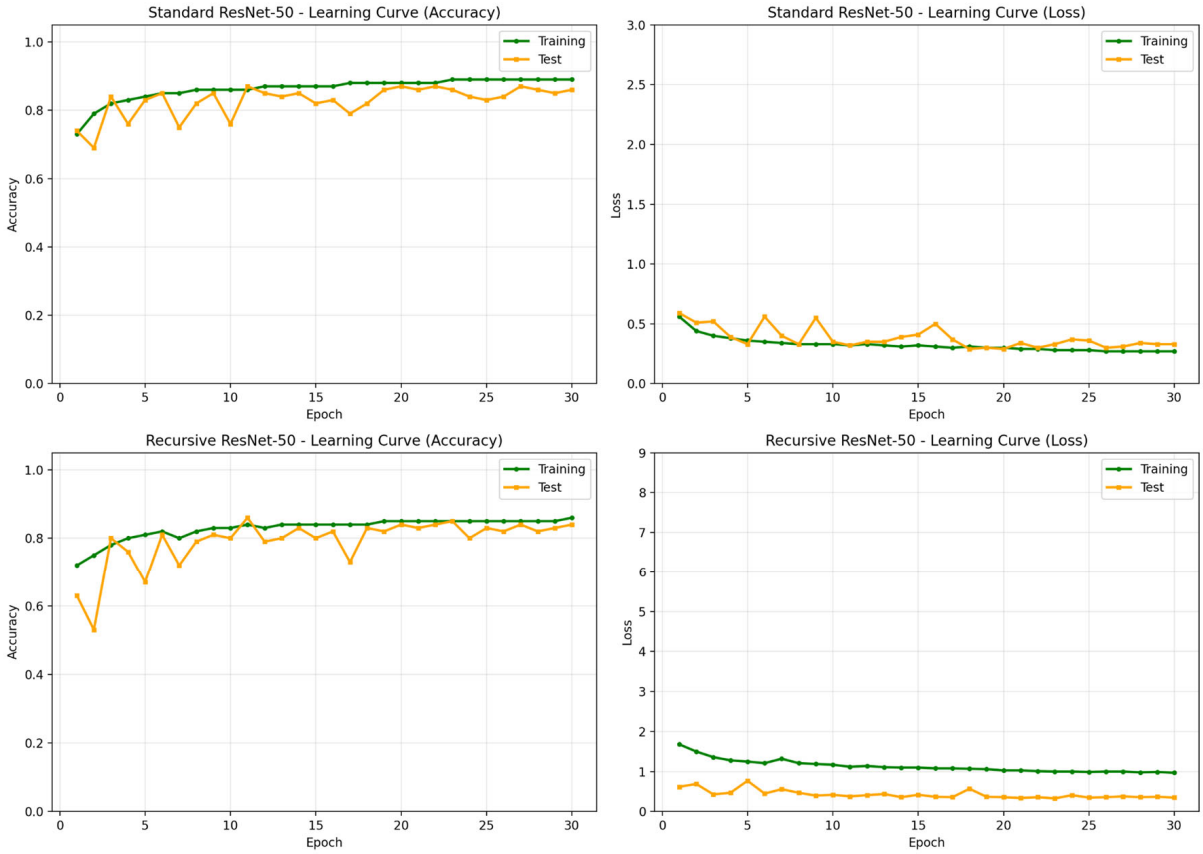

Figure 6. Learning Curves of Standard ResNet-50 and Recursive ResNet-50 Models.

The recall improvement comes with a trade-off in precision (85.36% vs. 93.59%), meaning that the recursive model produces more false positives. However, in melanoma screening, this trade-off is clinically favorable: a false positive leads to a follow-up biopsy (a low-risk, routine procedure), whereas a false negative a missed melanoma can result in delayed treatment and disease progression. The F1-score, which balances precision and recall, remains nearly identical (0.8695 vs. 0.8724), confirming overall performance parity. The ROC-AUC of 0.939 for the recursive model indicates excellent discrimination ability across thresholds.

Training time decreased by 56.7% (8155 s vs. 18,830 s), enabling faster experimental iterations and reduced energy consumption. This efficiency gain stems from the reduced parameter count and simpler per-cycle computations.

The ROC-AUC (Figure 7) of 0.939 for the recursive model (vs. 0.962 for standard) indicates excellent discrimination ability, confirming that the ranking quality of the predictions remains high despite the parameter reduction. Confusion matrix analysis (Figure 8) showed that the recursive model correctly classified 848 benign cases (vs. 944 for the standard model) and 886 malignant cases (vs. 817 for the standard model), explaining the recall improvement.

Results indicate that training time decreased by 56.7% (8155 s vs. 18,830 s), enabling faster experimental iterations and reduced energy consumption. This efficiency gain stems from the reduced parameter count and simpler per-cycle computations, although the sequential nature of recursive processing prevents perfect scaling with parameter reduction.

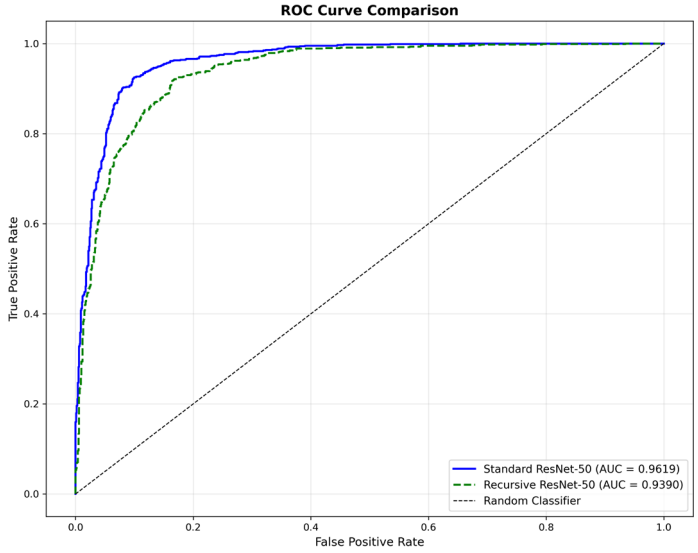

Figure 7. ROC Curve comparison between Standard Resnet-50 and Recursive ResNet-50.

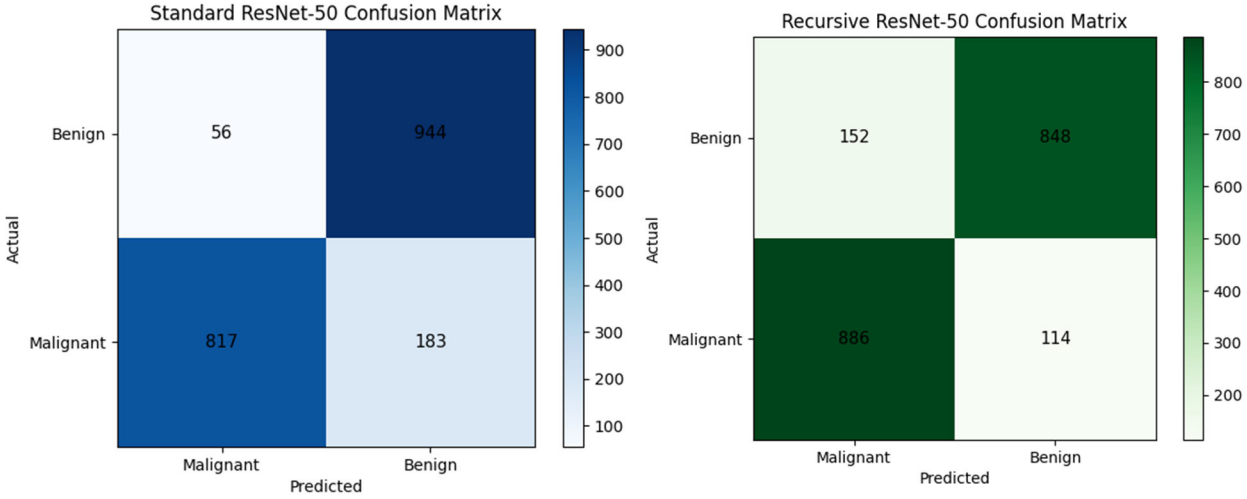

Figure 8. Confusion matrices of the Standard ResNet-50 and Recursive ResNet-50 models using the HAM10000 dataset.

4.2. Ablation Study: Effect of Recursive Cycles

Ablation Study on HAM10000 dataset:

To understand how recursive depth and training duration affect performance on the larger HAM10000 dataset, we conducted an ablation study comparing three recursive configurations (5, 6, and 7 cycles) against their corresponding standard baselines. Unlike small datasets where excessive parameters lead to overfitting, large datasets provide sufficient supervision for high-capacity models, making this ablation critical for determining when recursive architectures offer practical advantages.

Table 3 presents the complete results. We evaluated Tiny Recursive ResNet-50 with 5 cycles trained for 30 epochs, and with 6 and 7 cycles.

Table 3. Ablation study using HAM10000: Effect of recursive cycles and training epochs.

Cycles	Parameters	Accuracy	Precision	Recall	F1-Score	ROC-AUC
5	11.89M	0.8670	0.8536	0.8860	0.8695	0.9390
6	11.90M	0.8490	0.8591	0.8350	0.8469	0.9317
7	11.90M	0.8490	0.8591	0.8350	0.8469	0.9317

The ablation results demonstrate that, on a large-scale dataset such as HAM10000 (10,015 images), increasing recursive depth beyond a moderate number of cycles does not produce consistent performance gains. With only 5 recursive cycles, the model already achieves strong performance (86.70% accuracy and 0.939 ROC-AUC) while using fewer than 12 M parameters.

Increasing the number of cycles to 6 and 7 yields nearly identical classification performance, suggesting that recursive feature refinement converges rapidly when sufficient training data are available. Beyond this point, additional recursion primarily increases computational cost without improving discriminative ability. This behavior indicates a performance saturation effect, where deeper recursion does not translate into better generalization in data-rich scenarios.

These findings support the conclusion that, for large datasets, moderate recursion (5–7 cycles) provides the most favorable accuracy–efficiency trade-off. Deeper recursive configurations are therefore more appropriate for data-limited regimes, as further explored in the PH2 experiments.

Validation on PH2 dataset:

To assess generalization behavior under limited data, we validate all models on PH2 using 5-fold stratified cross-validation. Table 4 systematically examines the relationship between recursive depth and performance on PH2 (30 epochs).

Table 4. Validation on PH2 using 5-fold stratified cross-validation (mean ± std).

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Standard ResNet-50	0.935 ± 0.026	0.830 ± 0.055	0.850 ± 0.094	0.838 ± 0.065	0.939 ± 0.048
Recursive ResNet-50 (C = 5)	0.955 ± 0.019	0.908 ± 0.084	0.875 ± 0.079	0.886 ± 0.045	0.960 ± 0.038
Recursive ResNet-50 (C = 7)	0.945 ± 0.025	0.898 ± 0.088	0.825 ± 0.061	0.858 ± 0.062	0.959 ± 0.030
Recursive ResNet-50 (C = 10)	0.935 ± 0.026	0.852 ± 0.088	0.825 ± 0.061	0.836 ± 0.063	0.953 ± 0.030

Validation on the PH2 dataset using 5-fold stratified cross-validation reveals a clear performance trend across recursive configurations. The recursive refinement model with five cycles (C = 5) achieves the highest mean performance, yielding an accuracy of 0.955 ± 0.019 and an F1-score of 0.886 ± 0.045 compared to 0.935 ± 0.026 accuracy and 0.838 ± 0.065 F1-score for the standard ResNet-50 baseline. In addition to improved mean metrics, the C = 5 configuration exhibits reduced variability across folds, indicating enhanced prediction stability.

This behavior suggests that moderate recursive refinement improves feature representation quality without introducing excessive model complexity. The weight-sharing mechanism enables iterative representation enhancement while preserving parameter efficiency, which may reduce sensitivity to overfitting effects commonly observed in small datasets.

Increasing recursion depth beyond moderate levels does not yield consistent gains. For instance, deeper configurations (C = 7 and C = 10) show slightly lower mean performance and higher variance, with the C = 10 configuration reverting to baseline-level accuracy. These observations indicate that excessive recursive cycles may introduce diminishing returns, highlighting the importance of selecting an appropriate recursion depth.

4.3. Computational Efficiency and Resource Requirements

Table 5 provides comprehensive computational analysis, demonstrating that parameter reduction translates to practical efficiency gains across multiple metrics.

The recursive architecture demonstrates substantial efficiency improvements across all metrics except inference time. Model size reduction (49.4%) enables deployment on devices with limited storage, which is critical for mobile and edge applications. Training

time reduction (56.7%) accelerates research iteration and reduces energy costs, which is particularly valuable for institutions with limited computational budgets.

Table 5. Computational efficiency comparison (HAM10000, 30 epochs).

Metric	Standard ResNet-50	Tiny Recursive (5c)	Improvement
Parameters	23,512,130	11,894,351	−49.4%
Model Size (FP32)	89.7 MB	45.4 MB	−49.4%
Training Time	18,830 s (5.2 h)	8155 s (2.3 h)	−56.7%
Inference Time (per image)	24.3 ms	31.8 ms	+30.9%
Memory Usage (training)	3.2 GB	2.1 GB	−34.4%
FLOPs (forward pass)	4.09×10^9	2.87×10^9	−29.8%

Peak memory usage during training decreases by 34.4% (2.1 GB vs. 3.2 GB), enabling training on GPUs with less VRAM. This democratizes access to model development, allowing researchers with consumer-grade hardware to work with the architecture. FLOP reduction (29.8%) indicates lower computational complexity per forward pass, translating to reduced energy consumption, which is important for environmentally sustainable AI and battery-powered devices.

However, inference time increases by 30.9% (31.8 ms vs. 24.3 ms per image) due to sequential recursive cycles. For 5 cycles, this overhead remains acceptable: 31.8 ms enables processing of 31.4 images per second, sufficient for real-time applications. A typical medical imaging workflow (30–40 images) completes in approximately 1 s. The inference slowdown represents a trade-off inherent to recursive architectures: parameter sharing across time prevents full parallelization.

For deployment scenarios prioritizing model size and memory over inference speed such as mobile apps, embedded systems, or batch processing, the recursive architecture offers compelling advantages. For latency-critical applications requiring minimum per-sample inference time, standard architectures remain preferable despite higher resource requirements. Future work could explore adaptive recursive depth that terminates early for easy cases, reducing average inference time while maintaining accuracy on difficult examples.

5. Discussion

This study demonstrates that parameter efficiency can be effectively achieved through architectural design rather than relying solely on post hoc compression techniques. The proposed Tiny Recursive ResNet-50 reduces the number of parameters by approximately 49.4% while maintaining competitive performance on the HAM10000 dataset. This result suggests that increasing model capacity in terms of parameters is not always necessary to achieve strong predictive performance, particularly when alternative mechanisms such as recursive refinement are employed.

A key observation emerging from our experiments is the implicit regularization effect induced by parameter sharing. The validation results on the PH2 dataset, obtained using stratified 5-fold cross-validation, indicate not only improved mean performance but also reduced variability across folds compared to the standard ResNet-50. This behavior suggests that limiting the number of learnable parameters can help mitigate overfitting in small-data regimes. In this context, the recursive architecture provides a controlled increase in effective depth without introducing additional parameters, allowing the model to refine its representations while preserving generalization.

The relationship between dataset size and model capacity also becomes evident through our analysis. On the larger HAM10000 dataset, both the standard and recursive

architectures achieve strong performance, with the recursive model exhibiting a favorable trade-off between recall and precision. In contrast, on the smaller PH2 dataset, the benefits of parameter efficiency are more pronounced, as the recursive model achieves higher stability and improved overall performance. These findings suggest that the optimal architecture depends on the data regime, with parameter-efficient designs being particularly advantageous in scenarios with limited training data.

Another important aspect highlighted by our results is the role of recursive depth as a hyperparameter. The ablation study shows that moderate recursion (e.g., five cycles) provides the best balance between performance and stability. Increasing the number of recursive cycles beyond this point does not lead to consistent improvements and may even degrade performance. This indicates that excessive recursion can introduce diminishing returns, effectively increasing model complexity without proportional gains. Therefore, selecting an appropriate number of recursive iterations is essential and should be guided by the characteristics of the dataset.

From a clinical perspective, the observed trade-off between precision and recall is particularly relevant. The recursive model consistently achieves higher recall for melanoma detection, which is a desirable property in medical screening applications where missing a malignant case can have severe consequences. Although this improvement is accompanied by a reduction in precision, resulting in more false positives, such a trade-off is generally acceptable in clinical settings, where false positives can be resolved through additional diagnostic procedures. This highlights the importance of aligning model design with application-specific priorities.

In terms of computational efficiency, the proposed architecture offers substantial advantages, including reduced training time, lower memory consumption, and fewer floating-point operations. These improvements make the model more accessible for deployment in environments with limited computational resources. However, the sequential nature of recursive processing introduces a moderate increase in inference time. This trade-off reflects a fundamental characteristic of recursive architectures, where parameter sharing across time limits parallelization. Consequently, the choice between standard and recursive architectures should be guided by the specific requirements of the deployment scenario, particularly in terms of latency constraints.

Compared to traditional model compression techniques such as pruning, quantization, and knowledge distillation, the proposed approach integrates efficiency directly into the architectural design. This eliminates the need for complex multi-stage training pipelines and simplifies the overall development process. Furthermore, the recursive architecture remains compatible with existing compression methods, suggesting that additional efficiency gains could be achieved by combining both strategies.

Despite these promising results, several limitations should be acknowledged. First, the current study focuses on binary classification, and further investigation is needed to assess the behavior of recursive architectures in multi-class settings. Second, the experiments are limited to the domain of medical image classification, and the generalization of the proposed approach to other computer vision tasks, such as object detection or segmentation, remains to be explored. Finally, while the empirical results suggest an implicit regularization effect, a deeper theoretical understanding of this phenomenon would be valuable.

Future work should therefore consider extending recursive architectures to other backbone models, exploring adaptive mechanisms for dynamically selecting the number of recursive cycles, and evaluating the approach under more diverse datasets and task settings. In addition, investigating the interaction between recursive processing and other efficiency-oriented techniques may provide further insights into designing robust and scalable deep learning models.

6. Conclusions

In this work, we introduced Tiny Recursive ResNet-50, a parameter-efficient architecture that leverages recursive weight sharing to achieve substantial reductions in model complexity without compromising predictive performance. By reusing the same set of parameters across multiple refinement cycles, the proposed approach enables the network to achieve effective depth through iterative processing rather than structural expansion.

Experimental results on two datasets of different scales highlight the relevance of this design. On the large-scale HAM10000 dataset, the proposed model maintains competitive overall performance while significantly improving recall for melanoma detection, despite using nearly half the number of parameters. On the smaller PH2 dataset, the recursive architecture demonstrates improved stability and generalization, suggesting that parameter efficiency can play a beneficial role in low-data regimes.

These findings challenge the common assumption that increasing the number of parameters is the primary pathway to improved performance. Instead, they emphasize the importance of architectural design choices, particularly in scenarios where computational resources are limited or data availability is constrained. The recursive paradigm offers a compelling alternative by enabling a more efficient use of model capacity through iterative refinement.

From a practical perspective, the proposed approach provides a favorable trade-off between accuracy, computational cost, and memory requirements, making it suitable for deployment in resource-constrained environments such as mobile and embedded systems. While the increase in inference time due to sequential processing represents a limitation, it remains acceptable in many real-world applications, particularly those prioritizing model compactness and energy efficiency.

Overall, this work highlights the potential of recursive architectures as a viable direction for designing efficient deep learning models. Future research should further investigate adaptive mechanisms for controlling recursive depth, extend the approach to other architectures and tasks, and explore its integration with complementary efficiency techniques. Such efforts may contribute to the development of more scalable, accessible, and sustainable artificial intelligence systems.

Author Contributions: A.B. and M.A.S. jointly conceived and designed the Tiny Recursive ResNet-50 architecture. A.B. implemented the model and designed and conducted the HAM10000 experiments. A.B. and M.A.S. performed the statistical analysis and wrote the original draft of the manuscript. A.A. tested the model on the PH2 dataset and validated the results. M.A.S. supervised the research project, provided critical feedback on the experimental design, and reviewed and edited the manuscript. A.A. co-supervised the project, provided institutional resources and support, and reviewed the final manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data used in this study are publicly available. The HAM10000 dermoscopic image dataset is available from the Harvard Dataverse repository at: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/DBW86T (accessed on 22 March 2026). The PH2 dermoscopic image dataset is publicly available for research purposes and can be accessed at: https://www.fc.up.pt/addi/ph2%20database.html (accessed on 22 March 2026).

Acknowledgments: The authors acknowledge the use of AI-assisted tools for language editing and grammatical refinement of the manuscript. These tools were exclusively employed to enhance readability and textual clarity. All scientific contributions, including conceptualization, methodology, experimental design, data analysis, and interpretation of results, were performed entirely by the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

CNN	Convolutional Neural Network
TRM	Tiny Recursive Model
HAM10000	Human Against Machine with 10,000 Training Images
PH2	Dermoscopic Image Dataset from Hospital Pedro Hispano
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
ROC-AUC	Receiver Operating Characteristic–Area Under the Curve
FLOPs	Floating Point Operations
ReLU	Rectified Linear Unit
BN	Batch Normalization
GPU	Graphics Processing Unit

References

1. 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 (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
2. Tan, M.; Le, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the International Conference on Machine Learning (ICML), Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
3. Han, S.; Pool, J.; Tran, J.; Dally, W. Learning both Weights and Connections for Efficient Neural Network. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Montreal, QC, Canada, 7–12 December 2015; pp. 1135–1143.
4. Jacob, B.; Kligys, S.; Chen, B.; Zhu, M.; Tang, M.; Howard, A.; Adam, H.; Kalenichenko, D. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 2704–2713.
5. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2015**, arXiv:1503.02531. [CrossRef]
6. Zoph, B.; Le, Q.V. Neural Architecture Search with Reinforcement Learning. In Proceedings of the International Conference on Learning Representations (ICLR), Toulon, France, 24–26 April 2017.
7. Dehghani, M.; Gouws, S.; Vinyals, O.; Uszkoreit, J.; Kaiser, Ł. Universal Transformers. In Proceedings of the International Conference on Learning Representations (ICLR), New Orleans, LA, USA, 6–9 May 2019.
8. Zamir, A.R.; Wu, T.-L.; Sun, L.; Shen, W.B.; Shi, B.E.; Malik, J.; Savarese, S. Feedback Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1308–1317.
9. Liao, Q.; Poggio, T. Bridging the Gaps Between Residual Learning, Recurrent Neural Networks and Visual Cortex. *arXiv* **2016**, arXiv:1604.03640. [CrossRef]
10. George, D.; Lehrach, W.; Kansky, K.; Lázaro-Gredilla, M.; Laan, C.; Marthi, B.; Lou, X.; Meng, Z.; Liu, Y.; Wang, H.; et al. A generative vision model that trains with high data efficiency and breaks text-based CAPTCHAs. *Science* **2017**, *358*, eaag2612. [CrossRef] [PubMed]
11. Tschandl, P.; Rosendahl, C.; Kittler, H. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Sci. Data* **2018**, *5*, 180161. [CrossRef] [PubMed]
12. Mendonça, T.; Ferreira, P.M.; Marques, J.S.; Marcal, A.R.S.; Rozeira, J. PH2—A dermoscopic image database for research and benchmarking. In Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Buenos Aires, Argentina, 31 August–4 September 2013; pp. 5437–5440.
13. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861. [CrossRef]
14. 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 (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
15. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 6848–6856.
16. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. *arXiv* **2016**, arXiv:1602.07360.

17. Radosavovic, I.; Kosaraju, R.P.; Girshick, R.; He, K.; Dollár, P. Designing Network Design Spaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020; pp. 10428–10436.
18. Brock, A.; De, S.; Smith, S.L.; Simonyan, K. High-Performance Large-Scale Image Recognition Without Normalization. In Proceedings of the International Conference on Machine Learning (ICML), Virtual, 18–24 July 2021; pp. 1059–1071.
19. LeCun, Y.; Denker, J.S.; Solla, S.A. Optimal Brain Damage. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Denver, CO, USA, 27–30 November 1989; pp. 598–605.
20. Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; Graf, H.P. Pruning Filters for Efficient ConvNets. In Proceedings of the International Conference on Learning Representations (ICLR), Toulon, France, 24–26 April 2017.
21. Courbariaux, M.; Bengio, Y.; David, J.-P. BinaryConnect: Training Deep Neural Networks with binary weights. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Montreal, QC, Canada, 7–12 December 2015; pp. 3123–3131.
22. Nagel, M.; Fournarakis, M.; Amjad, R.A.; Bondarenko, Y.; van Baalen, M.; Blankevoort, T. A White Paper on Neural Network Quantization. *arXiv* **2021**, arXiv:2106.08295. [CrossRef]
23. Krishnamoorthi, R. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv* **2018**, arXiv:1806.08342. [CrossRef]
24. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge Distillation: A Survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. [CrossRef]
25. Denton, E.L.; Zaremba, W.; Bruna, J.; LeCun, Y.; Fergus, R. Exploiting Linear Structure Within Convolutional Networks for Efficient Evaluation. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Montreal, Canada, 8–13 December 2014; pp. 1269–1277.
26. Kim, Y.-D.; Park, E.; Yoo, S.; Choi, T.; Yang, L.; Shin, D. Compression of Deep Convolutional Neural Networks for Fast and Low Power Mobile Applications. In Proceedings of the International Conference on Learning Representations (ICLR), San Juan, Puerto Rico, 2–4 May 2016.
27. Liang, M.; Hu, X. Recurrent convolutional neural network for object recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3367–3375.
28. Wang, F.; Jiang, M.; Qian, C.; Yang, S.; Li, C.; Zhang, H.; Wang, X.; Tang, X. Residual Attention Network for Image Classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 3156–3164.
29. Schlemper, J.; Oktay, O.; Schaap, M.; Heinrich, M.; Kainz, B.; Glocker, B.; Rueckert, D. Attention gated networks: Learning to leverage salient regions in medical images. *Med. Image Anal.* **2019**, *53*, 197–207. [CrossRef] [PubMed]
30. Shi, X.; Chen, Z.; Wang, H.; Yeung, D.-Y.; Wong, W.-K.; Woo, W.-C. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Montreal, QC, Canada, 7–12 December 2015; pp. 802–810.
31. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-End Object Detection with Transformers. In Proceedings of the European Conference on Computer Vision (ECCV), Glasgow, UK, 23–28 August 2020; pp. 213–229.
32. Shen, Z.; Zhang, M.; Zhao, H.; Yi, S.; Li, H. Efficient Attention: Attention with Linear Complexities. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 3–8 January 2021; pp. 3531–3539.