

# An Optimized Deep Neural Network Model for Image Classification in Resource-constrained Environments

**Raafi Careem\***

Department of Computer Science & Informatics, Uva Wellassa University, Badulla, Sri Lanka

E-mail: [mraafi@gmail.com](mailto:mraafi@gmail.com)

ORCID iD: <https://orcid.org/0009-0001-8632-4019>

\*Corresponding author

**Md Gapar Md Johar**

Software Engineering and Digital Innovation Centre, Management and Science University, Shah Alam, Malaysia

E-mail: [mdgapar@msu.edu](mailto:mdgapar@msu.edu)

ORCID iD: <https://orcid.org/0000-0001-7986-3834>

Received: 10 October 2025; Revised: 10 November 2025; Accepted: 06 January 2026; Published: 08 April 2026

**Abstract:** Advances in deep learning have highlighted the need for models tailored for deployment in resource-constrained environments (RCEs), where memory and processing limitations present significant challenges, such as those found in mobile devices, Internet of things (IoT) devices, and embedded systems. This paper introduces GRMobiNet, a novel deep neural network (DNN) model designed to address these challenges in image classification tasks by balancing computational complexity with model accuracy in RCE settings. The model focuses on key performance goals inspired by previous state-of-the-art models, aiming to achieve a better balance between complexity and accuracy. These goals include reducing the model's computational complexity to fewer than 4 million parameters, limiting memory usage to under 16 megabytes, and achieving an accuracy greater than 80%. By meeting these objectives, GRMobiNet enhances both the effectiveness and efficiency of deep neural network deployment in RCE settings. GRMobiNet builds upon MobileNet as its baseline, incorporating advanced techniques such as depthwise separable convolutions, compound scaling, global average pooling, and quantization to optimize performance. Trained on ImageNet-10, a subset of ImageNet-1K, the model underwent rigorous performance evaluation. Experimental results demonstrate that GRMobiNet achieves its performance objectives, with a computational complexity of 3.2 million parameters, memory utilization of 12.6 megabytes, and a prediction accuracy of 92%, validating its suitability for RCEs. This research presents a scalable framework for balancing accuracy and computational efficiency, with significant implications for RCE devices. In future work, GRMobiNet will be tested on commercially available RCE mobile devices using real-world images to assess its practicality and evaluate its performance in terms of accuracy, confidence, and inference time for image classification in real-world scenarios.

**Index Terms:** Deep Neural Network, GRMobiNet, Image Classification, Model Deployment, Optimization, Resource-constrained Environment.

## 1. Introduction

Deep neural network (DNN) is a machine learning (ML) approach based on artificial neural networks (ANN). DNN has proven to be effective in numerous types of applications, such as computer vision [1, 2], natural language processing [3], autonomous vehicle [4], health care [5], agriculture [6], and more in recent years. The most widespread DNN application in computer vision is image classification [7]. However, the deployment of DNNs for image classification on resource-constrained environments (RCE) is challenging, due to the limited computational as well as energy resources of these devices [8]. The high computational demands of DNNs result in long inference times, high energy consumption, and increased heat dissipation, which can lead to negative impacts on battery life and performance in RCE devices.

Image classification is a task in computer vision that involves categorizing an image into one or more predefined classes based on its visual content [9]. This process typically involves training a deep learning model, most commonly a

convolutional neural network (CNN), to identify patterns and features in images that distinguish between different classes. The model is trained on a labeled dataset of images, and once trained, it can classify new images into one of the predefined categories. Deep neural networks, the CNNs, have achieved state-of-the-art performance on image classification datasets [10-14]. However, deploying DNNs on RCE devices such as mobile phones and embedded systems (refer Table 1) can be a challenging task due to the limited computational resources available on these devices [15].

RCE are characterized by constraints in terms of memory and processing power, and these limitations can lead to difficulties in deploying DNNs, which are typically large and computationally intensive models. Table 1 presents a list of various RCE devices with their configuration, type of model, CPU power in GHz and memory size in GB. The table shows that different devices have varying computational resources, with some devices having more memory and faster central processing unit (CPU) speeds than others.

Table 1. Typical Resource- constrained Devices (CPU: 1.43- 2.84GHz, MEM: 1-8GB)

| RCE Device Name          | CPU Model                 | CPU Power (GHz)  | Memory (GB) |
|--------------------------|---------------------------|------------------|-------------|
| Raspberry Pi -4B         | Broadcom -BCM2711         | 1.50 (quad-core) | 2.0 - 8.0   |
| NVIDIA - Jetson Nano     | NVIDIA Carmel ARMv -8.2   | 1.43 (quad-core) | 4.0         |
| Google - Coral Dev Board | NXP i.MX - 8M             | 1.50 (quad-core) | 1.0 - 4.0   |
| Google - Pixel           | Qualcomm Snapdragon - 821 | 2.15 (quad-core) | 4.0         |
| iPhone - XR              | Apple A12 - Bionic        | 2.49 (hexa-core) | 3.0         |
| Google - Pixel 3         | Qualcomm Snapdragon - 845 | 2.50 (octa-core) | 4.0         |
| Google - Pixel 2         | Qualcomm Snapdragon - 835 | 2.35 (octa-core) | 4.0         |
| Google Pixel 4           | Qualcomm Snapdragon - 855 | 2.84 (octa-core) | 6.0         |
| Xiaomi Mi -10            | Qualcomm Snapdragon - 865 | 2.84 (octa-core) | 8.0         |
| Samsung S10              | Exynos 9820               | 2.73 (octa-core) | 8.0         |
| Huawei P30 Pro           | Kirin 980                 | 2.60 (octa-core) | 9.0         |

Sources: [14, 16-18]

The challenges of deploying DNNs in RCE include the need to reduce memory usage and computation time, while maintaining or improving accuracy. To overcome these challenges, various optimization techniques, such as depthwise separable convolution and compound scaling, and models such as MobileNet and EfficientNet have been proposed in the literature. These techniques and models aim to reduce the computational and memory demands of deep neural networks while maintaining or improving their accuracy.

In light of the challenges associated with deploying DNNs in RCEs, this research focuses on developing an optimized DNN model specifically tailored for such settings. The proposed model, GRMobiNet, builds on the foundations laid by techniques like depthwise separable convolution and compound scaling, as seen in models such as MobileNet and EfficientNet. However, GRMobiNet extends these techniques by addressing the unique constraints of RCEs, such as limited memory and processing power, while still aiming to achieve high accuracy in image classification tasks. This research contributes to bridging the gap between high-performance DNNs and the practical limitations of RCEs, offering a novel solution that balances computational efficiency and accuracy.

Table 2. Comparison of benchmark DNN models for image classification on the imagenet dataset

| DNN Model         | No. of Parameters of the model (m) | Model Size (MB) | Time (ms) per inference step (CPU) | Accuracy (%) | Related references |
|-------------------|------------------------------------|-----------------|------------------------------------|--------------|--------------------|
| MobileNet         | 4.3                                | 16              | 22.6                               | 70           | [16, 17, 19-21]    |
| EfficientNetB0    | 5.3                                | 29              | 46                                 | 77           | [14, 16]           |
| EfficientNetB1    | 7.9                                | 31              | 60.2                               | 79           | [14, 16, 22]       |
| DenseNet121       | 8.1                                | 33              | 77.1                               | 75           | [16, 23]           |
| InceptionV3       | 23.9                               | 92              | 42.2                               | 79           | [16, 24]           |
| ResNet50          | 25.6                               | 98              | 58.2                               | 75           | [16, 25]           |
| InceptionResNetV2 | 55.9                               | 215             | 130.2                              | 80           | [16, 26]           |
| ResNet152V2       | 60.4                               | 232             | 107.5                              | 78.0         | [16, 25, 27]       |
| EfficientNetB7    | 66.7                               | 256             | 1578.9                             | 84.3         | [14, 16]           |

When developing a DNN model focused on efficiency, particularly for deployment on RCE devices, several factors must be carefully considered, including the model's architecture, size (memory requirements), and computational demands (number of parameters). These factors directly impact the performance and accuracy of the

model, as well as its ability to run efficiently on RCEs. For instance, to achieve high accuracy levels, DNNs often require large numbers of parameters, which in turn increases the computational and memory requirements, making deployment on resource-constrained devices challenging. The subsequent Table 2 illustrates these considerations in detail.

Table 2 presents a comparison of several benchmark DNN models for image classification on the ImageNet dataset. The models are evaluated based on four key attributes: the number of parameters, model size (in MB), inference time per step on a CPU (in milliseconds), and classification accuracy (in percentage). The models range from lightweight architectures, like MobileNet, to more complex ones, such as EfficientNetB7 and InceptionResNetV2, highlighting the trade-off between computational efficiency and classification accuracy.

As shown in the table, there is a clear trade-off between the different DNN models. The most accurate models, EfficientNetB7 and InceptionResNetV2, have large model sizes (256 MB and 215 MB), a high number of parameters (66.7M and 55.9M), and longer inference times (1578.9 ms and 130.2 ms). However, they achieve high accuracy (84.3% and 80%), correctly recognizing the main object in a large percentage of images. These models are computationally demanding and require significant processing power and memory capacity.

In contrast, models like MobileNet and EfficientNetB0 have smaller model sizes (16 MB and 29 MB), fewer parameters (4.3M and 5.3M), and faster inference times (22.6 ms and 46 ms). They are designed as lightweight models for RCEs. However, they sacrifice some accuracy (70.4% and 77.1%) to achieve these efficiency gains.

This leads to the primary research gap identified in this study: the lack of optimization models specifically tailored for DNN deployment on RCEs for image classification. While existing models offer some optimization, they do not adequately address the unique constraints and limitations of RCEs, such as limited memory and processing power. Therefore, there is a clear need to develop DNN models that effectively balance memory utilization, computational complexity, and accuracy to enable successful image classification on RCEs.

The proposed DNN model aims to address this gap by targeting specific performance objectives. It seeks to reduce computational complexity by limiting the number of parameters to fewer than 4 million and minimizing memory utilization to below 16 megabytes [16, 17, 19-21], while achieving a minimum accuracy of 80% [16, 26]. These goals align with the specific constraints commonly found in RCEs, including memory (1-8 GB) and processing power (1.43-2.84 GHz).

The proposed DNN model aims to assist the deployment of high-quality image classification systems on RCEs by striking a balance between accuracy and efficiency. This is essential due to the limited processing capabilities of RCEs, such as mobile devices. To effectively address the challenges of DNN deployment on RCEs, it is crucial to develop strategies that optimize model accuracy, reduce computational complexity, and minimize memory usage. The ultimate goal is to enable high-quality image classification on RCEs by creating a DNN model tailored to meet the specific needs and constraints of these devices.

The rest of this paper is organized as follows. The next section reviews the literature related to the design and development of lightweight DNN models for image classification applications. Section III details the methodology, including the methods and materials used in this research to achieve the objectives. Section IV presents the experimental results and discussions, including performance evaluation and a comparative analysis of the GRMobiNet model. The final section provides the conclusions and outlines future work for this research.

## 2. Related Works

In recent years, various DNN models and optimization techniques have been developed to address the challenges of deploying models on RCEs. These approaches aim to balance computational efficiency and accuracy. However, despite their innovations, each model and technique has limitations that have informed the development of GRMobiNet, a model specifically tailored for RCEs, such as mobile devices with limited memory and processing power.

MobileNet model [17, 18, 28-30] is one of the pioneering attempts to optimize DNNs for devices with limited computing power. The model achieved significant computation reduction while retaining reasonable accuracy, making it a preferred choice for mobile applications. However, its lower accuracy compared to other DNNs, coupled with the reliance on trial-and-error optimization, highlights its performance limitations, especially in more resource-constrained settings.

ResNet50 and ResNet152V2 [24, 26] advanced the field by introducing deep networks with skip connections to combat the vanishing gradient problem. While these models offer state-of-the-art accuracy and effective feature learning, their complex architectures make them resource-intensive, prone to overfitting, and in need of regularization techniques. These drawbacks are especially problematic in RCEs, where computational and memory resources are limited.

InceptionV3 and InceptionResNetV2 [24, 26, 31] pushed the boundaries of efficient computation by combining varied filter sizes and residual connections. While these models are versatile and offer improved accuracy, they are difficult to train and fine-tune, suffer from a loss of image details, and impose high memory demands. These issues limit their applicability in environments where memory is a critical constraint.

DenseNet121 [23, 32] offers a unique approach through dense connectivity, improving gradient flow and making it suitable for transfer learning with fewer network parameters. However, the model's high memory consumption and

computational cost, along with potential accuracy loss due to bottleneck layers, restrict its deployment in RCEs, where both memory and processing power are constrained.

EfficientNetB1 [14, 22] represents a significant advancement in balancing accuracy and computational efficiency through compound scaling. While EfficientNetB1 achieves high accuracy and efficient training, its complex architecture increases computational cost, making it less suitable for RCEs.

In addition to the models specified above, various optimization techniques have been developed to further reduce the computational and memory demands of DNN models. Network Pruning, explored by Li et al. in [33] and Molchanov et al. in [34], reduces model size by removing less significant connections, leading to faster inference. However, this often comes at the cost of reduced accuracy and increased training complexity, making it less ideal for models that require high accuracy in RCEs. Weight Quantization, studied by Polino et al. [35] and Yang et al. [36], reduces the memory footprint and power consumption. Yet, the trade-off in accuracy and performance poses challenges for applications that require precise predictions. Knowledge Distillation, proposed by Yim et al. in [37] and Fu et al. in [38], transfers knowledge from a complex DNN to a simpler model, resulting in efficient models with faster inference. However, the potential loss of information and the increased computational cost can negate the benefits, especially in RCEs. Architecture Search, automated by Liu et al. [39] and Elsken et al. [40], attempts to discover optimal DNN architectures for specific tasks. While this technique enhances performance and efficiency, its high computational cost and the risk of overfitting present significant challenges in RCEs. Depthwise Separable Convolution, introduced by Chollet in [41] and Factorization of Convolution, explored by Wu et al. in [42], aim to reduce model size and improve efficiency. However, both techniques can result in reduced accuracy and increased model complexity, limiting their applicability in environments where maintaining high accuracy is critical. Residual Connections and Dense Connections, proposed by Qassim et al. [43] and Huang et al. [44], respectively, have been shown to enhance accuracy and improve gradient flow. However, the increased model complexity and memory usage associated with these techniques create barriers to their use in RCEs.

In light of these challenges, the GRMobiNet model has been designed to specifically address the limitations identified in these existing approaches. By integrating a baseline model such as MobileNet with a novel compound scaling and global average pooling technique, GRMobiNet achieves a superior balance between accuracy and computational complexity. Unlike existing models, GRMobiNet has been meticulously optimized for deployment in RCEs, ensuring that it meets the reduced memory and processing constraints of devices with limited resources. Additionally, as discussed previously, GRMobiNet's focus on reducing computational complexity to fewer than 4 million parameters and minimizing memory utilization below 16 megabytes differentiates it from prior models, making it a viable solution for real-world applications where resource efficiency is paramount.

### 3. Materials and Methods

#### 3.1. GRMobiNet Model

The research methodology has been developed to address the objective of creating an optimized DNN for RCEs and evaluating its performance. Accordingly, this paper proposes GRMobiNet, an optimized, efficiency-focused DNN model for image classification in RCEs. The development of GRMobiNet was grounded in the transfer learning approach [45], a widely adopted methodology in deep learning that accelerates model training by leveraging the knowledge from pre-trained models. Specifically, the weights of pre-trained DNN models served as the foundation for training the GRMobiNet model. These pre-trained weights capture a wealth of learned features—such as edges, textures, and shapes—that were extracted from large, diverse image classification datasets like ImageNet-1K. By incorporating these pre-learned patterns, GRMobiNet was able to capitalize on the general knowledge encoded within the base model, significantly accelerating its own learning process.

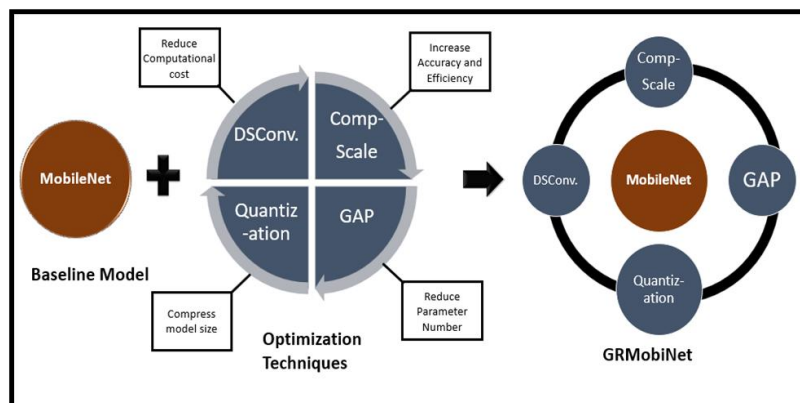


Fig.1. Outline of GRMobiNet model

This transfer learning approach dramatically reduces the training time of the new model, as the pre-trained weights already contain useful features learned from large datasets. Leveraging this knowledge allows the model to be fine-tuned on a smaller, task-specific dataset, rather than requiring the resource-intensive process of training from scratch. As part of this strategy, GRMobiNet uses the pre-trained MobileNet as a baseline model, transferring its weights to the new architecture. Moreover, as shown in Figure 1, the baseline model, MobileNet, is further enhanced with four advanced optimization techniques: depthwise separable convolution (DSConv), compound scaling (ComSc), global average pooling (GAP), and quantization. These integrations collectively contribute to the development of the GRMobiNet model.

Figure 1 demonstrates the GRMobiNet model, which is made up of two key components: a baseline model and optimization techniques. The details of each component are provided in the following subsections.

#### A. Baseline Model

In this research, MobileNet [17, 18, 28, 30, 46] served as the baseline model for developing GRMobiNet. MobileNet, known for its lightweight architecture and efficiency, was chosen due to its proven suitability for deployment in RCEs. Its design principles, including depthwise separable convolutions, effectively reduce computational complexity and model size without significant compromises in accuracy. By building on MobileNet, GRMobiNet incorporates and enhances these foundational features, integrating additional optimization techniques such as compound scaling, global average pooling, and quantization to further improve performance. This strategic selection of MobileNet as the baseline ensured a strong starting point for creating a highly efficient and scalable model tailored to RCEs.

#### B. Optimization Techniques

##### a. Depthwise Separable Convolution

Depthwise Separable Convolution (DSConv) [41, 46] is a critical optimization technique integrated into the proposed GRMobiNet model. This technique significantly enhances the model's efficiency, making it well-suited for deployment in RCEs.

Traditional convolutional layers in deep neural networks are computationally intensive, requiring substantial processing power and memory. DSConv, however, decomposes the standard convolution operation into two separate stages: depthwise convolution and pointwise convolution. This separation reduces both the number of parameters and the computational load, leading to a more efficient model architecture without sacrificing performance.

- **Depthwise Convolution:** This step applies a single convolutional filter to each input channel individually, rather than combining all channels. Each filter processes its corresponding channel independently, producing feature maps that capture spatial information within each channel separately.
- **Pointwise Convolution:** After depthwise convolution, a  $1 \times 1$  convolution is applied across the depthwise-convoluted feature maps. This operation integrates spatial information from the depthwise convolution with the depthwise feature map information, effectively merging the information across all channels.

In the context of GRMobiNet, DSConv plays a pivotal role in achieving the model's efficiency objectives. By incorporating DSConv, GRMobiNet maintains a compact model size and reduces inference time—both of which are critical for RCE applications. Here's how DSConv contributes to GRMobiNet:

- **Model Size Reduction:** By decomposing the convolution operation into depthwise and pointwise convolutions, DSConv significantly reduces the total number of parameters. This reduction leads to a smaller model size, which is especially beneficial for deployment on devices with limited storage and memory.
- **Computational Efficiency:** The separation of the convolution operations reduces the computational complexity of the model. This makes GRMobiNet faster in terms of inference, which is crucial for real-time applications in RCEs.
- **Maintained Accuracy:** Despite the reduction in parameters and computational demands, DSConv does not compromise model accuracy. GRMobiNet leverages DSConv's ability to efficiently capture essential features, ensuring high predictive performance is maintained.

##### b. Compound Scaling

Compound Scale (CompScale) [14, 22, 47] is a key optimization technique in GRMobiNet, designed to enhance efficiency and accuracy for deployment in RCEs. By uniformly scaling the depth, width, and resolution of the network, CompScale ensures balanced resource utilization and high performance. This approach is particularly valuable for mobile devices and embedded systems with limited computational and memory resources.

Traditional scaling methods focus on a single dimension, such as increasing depth (layers), width (channels), or resolution (input size), often leading to inefficiencies. CompScale addresses this limitation by scaling all three dimensions proportionally, achieving an optimal balance between model size, computational complexity, and accuracy. Depth scaling allows the network to learn complex features, width scaling captures fine-grained patterns, and resolution

scaling improves recognition of intricate details.

In GRMobiNet, CompScale ensures efficient resource utilization, allowing the model to achieve high prediction accuracy while maintaining computational efficiency. This balanced approach enables GRMobiNet to meet its performance objectives, including minimizing memory usage, reducing computational complexity, and achieving scalability for diverse RCEs. With CompScale, GRMobiNet demonstrates its suitability for real-world deployment across a range of devices, from low-power embedded systems to mobile platforms.

#### c. Global Average Pooling

Global Average Pooling (GAP) [18, 21, 28, 29] is a crucial optimization technique employed in the GRMobiNet model to enhance efficiency and performance, particularly for image classification tasks in RCEs. GAP replaces traditional fully connected layers with a lightweight mechanism, reducing parameters and computational load while maintaining high accuracy. By averaging the spatial dimensions of feature maps into a single value, GAP eliminates dense connections, minimizing overfitting and improving model generalization.

In GRMobiNet, GAP plays a pivotal role in reducing model complexity. By cutting down the number of parameters, it decreases the model size and memory usage, making GRMobiNet ideal for RCEs such as mobile devices and embedded systems. Furthermore, GAP accelerates inference times, enabling real-time processing, which is essential for applications requiring quick decision-making. Despite the reduced complexity, GAP captures global spatial information, preserving the image's contextual features and ensuring robust classification performance.

The integration of GAP aligns with GRMobiNet's goal of achieving a balance between efficiency and effectiveness. It significantly reduces computational demands and storage requirements, ensuring that the model is scalable and deployable across a wide range of devices. By leveraging GAP, GRMobiNet maintains its high accuracy while addressing the challenges of RCEs, reinforcing its suitability for real-world applications.

#### d. Quantization

Quantization [35, 36, 48] is a key optimization technique integrated into GRMobiNet, designed to enhance its efficiency by reducing the precision of weights and activations. This method is particularly critical for deployment in RCEs, where computational and memory resources are limited. By reducing the bit-width representation of parameters, quantization significantly minimizes the model's size and computational load while maintaining high performance.

In GRMobiNet, quantization involves converting 32-bit floating-point weights and activations to 8-bit integers. This reduction offers substantial benefits:

- **Reduced Model Size:** Fewer bits per parameter result in a smaller overall model, which is vital for devices with limited storage.
- **Faster Inference Speed:** Low-bit-width computations accelerate inference, critical for real-time applications.
- **Energy Efficiency:** Reduced computational complexity lowers power consumption, extending battery life on mobile and embedded devices.

#### e. Justification for Selected Optimization Techniques in GRMobiNet

GRMobiNet integrates a suite of carefully selected above optimization techniques to address the performance and efficiency challenges associated with RCEs. Each technique contributes uniquely to reducing the model's computational cost and memory footprint while preserving, or even improving, classification accuracy.

- Depthwise Separable Convolutions replace standard convolutions with a more efficient decomposition into depthwise and pointwise operations, significantly reducing parameter count and computation.
- Compound Scaling systematically scales model depth, width, and input resolution in a balanced manner, promoting efficiency across various hardware settings.
- Global Average Pooling substitutes dense fully connected layers with spatial averaging, which lowers model complexity and enhances generalization.
- Quantization reduces the precision of model weights and activations (e.g., from 32-bit floats to 8-bit integers), leading to faster inference and reduced memory consumption with minimal accuracy loss.

A summary of these techniques along with their implementation and impact within GRMobiNet is provided in Table 3.

As shown in Table 3, each optimization technique contributes to GRMobiNet's overall performance in a distinct and complementary way. Depthwise Separable Convolutions significantly reduce computational cost without sacrificing accuracy. Compound Scaling ensures balanced and scalable architecture across different resource levels. Global Average Pooling simplifies the model while retaining essential spatial features. Quantization further reduces memory and latency, making the model suitable for deployment on edge devices with strict hardware limitations. Together, these techniques form a cohesive optimization strategy that enables GRMobiNet to achieve high accuracy with low resource usage.

Table 3. Summary of optimization techniques in GRMobiNet

| Optimization Technique                    | Purpose                                      | Implementation in GRMobiNet                                                      | Effect                                                                   |
|-------------------------------------------|----------------------------------------------|----------------------------------------------------------------------------------|--------------------------------------------------------------------------|
| Depthwise Separable Convolution (DSCConv) | Reduce parameters and computational cost     | Decomposes standard convolutions into depthwise and pointwise operations         | Smaller model size, faster inference, accuracy maintained                |
| Compound Scaling (ComScale)               | Balanced scaling of model complexity         | Uniformly scales depth, width, and input resolution using compound coefficients  | Improved efficiency, scalability across device configurations            |
| Global Average Pooling (GAP)              | Reduce overfitting and model complexity      | Replaces fully connected layers with average pooling over feature maps           | Fewer parameters, better generalization, retained spatial information    |
| Quantization                              | Reduce memory usage and accelerate inference | Converts floating-point weights/activations to lower-precision (e.g., 8-bit int) | Smaller memory footprint, faster inference, minimal accuracy degradation |

### 3.2. Architecture of GRMobiNet

The architecture of the GRMobiNet is systematically constructed to utilize the above advanced optimization techniques and baseline model while ensuring efficiency and high performance for image classification tasks in RCE devices. The architecture as shown in Figure 2, is composed of seven primary components, input image, baseline model, compound scaling block, pooling block, classification block, compression block and output prediction block. The details of the each of the components are described as follows:

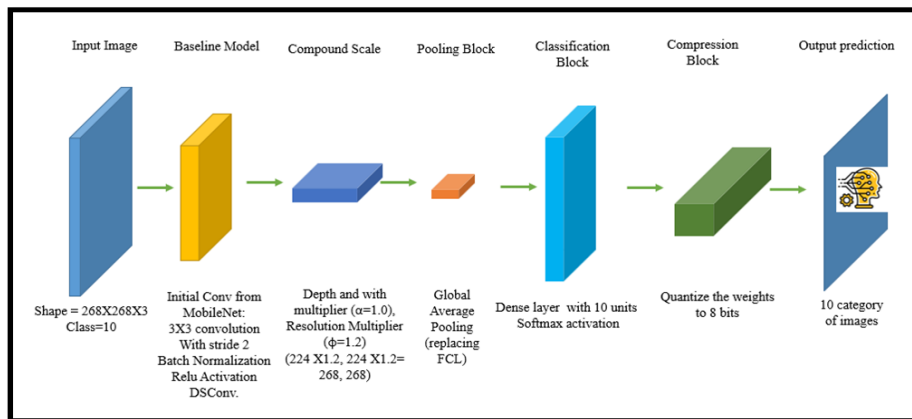


Fig.2. Architecture of GRMobiNet model

#### a. Input Image

The GRMobiNet model processes images with an input shape of 268 x 268 x 3 for 10 distinct classes, optimized for efficiency and detail capture in RCEs. The dimensions 268 x 268 result from scaling the standard size of 224 x 224 using a compound scaling factor ( $\phi = 1.2$ ), enabling the model to capture finer visual details crucial for distinguishing complex features while balancing computational demands. The three-color channels (RGB) capture the full color spectrum, allowing the model to leverage distinctive color patterns for feature extraction, such as recognizing a goldfish's orange hue or a parrot's vibrant feathers. GRMobiNet is designed to classify images into 10 categories (e.g., aeroplane, balloon, dog, horse) using a softmax layer that outputs probability scores, with the highest score determining the predicted class. This input configuration ensures GRMobiNet achieves high accuracy, efficient processing, and adaptability for deployment on devices with limited computational resources.

#### b. Baseline Model

The baseline model for GRMobiNet, derived from the initial layers of MobileNet, plays a vital role in the overall architecture. The 3x3 convolution layer with a stride of 2, followed by batch normalization and ReLU activation, provides a strong and efficient foundation for feature extraction. The inclusion of depthwise separable convolutions further enhances the model's efficiency, making it capable of delivering high performance with reduced computational demands. These components ensure that GRMobiNet can effectively process input images, extract relevant features, and build upon them in deeper layers, ultimately leading to a powerful and efficient model for image classification tasks.

#### c. Compound Scale

Compound scaling is a vital optimization technique used in GRMobiNet to enhance the model's efficiency and performance. This method simultaneously scales the depth, width, and resolution of the network, achieving a balanced and efficient model. By applying compound scaling, GRMobiNet leverages the advantages of increasing model

capacity while maintaining computational efficiency. The compound scaling factors used in GRMobiNet are the depth and width multiplier ( $\alpha=1.0$ ) and the resolution multiplier ( $\phi=1.2$ ). These values were obtained through an empirical study with a trial and error method to find the best configuration for the GRMobiNet model.

#### *d. Pooling Block*

The pooling block in GRMobiNet uses Global Average Pooling (GAP), a technique that enhances the model's efficiency and performance by replacing traditional fully connected layers (FCL). GAP reduces the number of parameters, lowering computational cost and memory usage, which is particularly beneficial for deployment in RCEs. It also helps prevent overfitting by eliminating the dense connections typical of FCLs, thus improving the model's generalization capability. Additionally, GAP improves interpretability by providing a direct correlation between high-level features and the final classification output. This makes GRMobiNet more efficient, with simplified architecture, faster training, and preserved spatial information, leading to more accurate predictions. The integration of GAP makes GRMobiNet a lightweight and efficient deep learning model, ideal for tasks in environments with limited resources while maintaining high performance.

#### *e. Classification Block*

The classification block in GRMobiNet is essential for transforming high-level features extracted by earlier layers into class probabilities for final predictions. It consists of a dense layer with 10 units, each corresponding to one of the 10 target classes, and uses the softmax activation function to output a probability distribution. The dense layer aggregates features from previous layers, generating a score for each class, which reflects the model's confidence in the presence of that class in the input image. The softmax function then normalizes these scores into a probability distribution, ensuring they sum to 1. This process enables GRMobiNet to provide interpretable predictions, where each probability indicates the model's confidence in each class. Key benefits of this design include interpretability, allowing users to easily understand the model's predictions, confidence measurement, which is useful for critical applications, and training efficiency, which optimizes multi-class classification performance.

#### *f. Compression Block*

The compression block in GRMobiNet plays a crucial role in enhancing the model's efficiency and performance, particularly for deployment on resource-constrained devices. It employs 8-bit quantization to reduce the memory and computational requirements without significantly affecting accuracy. Typically, deep learning models use 32-bit floating-point weights, which offer high precision but consume substantial memory and computational resources. In GRMobiNet, these weights are converted into 8-bit integers, effectively reducing the model's memory footprint by a factor of four, making it suitable for devices with limited storage and processing power. This quantization leads to several benefits: it reduces model size, enhancing memory efficiency and enabling storage savings, while also improving computational efficiency, resulting in faster inference and reduced latency. Additionally, the lower computational burden translates to energy savings, contributing to extended battery life, which is essential for mobile and portable devices. The quantization process maintains a balance between precision and performance, ensuring high accuracy despite reduced bit-width, and allows for broader deployment across a range of devices, from smartphones to embedded systems. This compression approach makes GRMobiNet scalable, efficient, and well-suited for real-time, resource-limited applications.

#### *g. Output Prediction Block*

The output prediction block in GRMobiNet is the final stage of the model's architecture, responsible for producing the classification results by assigning one of the ten predefined categories to the input image. Using the softmax activation function, it generates a probability distribution over the ten categories and selects the category with the highest probability as the final class label. This design ensures the predictions are both accurate and interpretable. By mapping complex features to a clear output, it simplifies the model's results, making them easily understood and actionable. Additionally, the output block provides insight into the model's confidence in its predictions, enhancing interpretability and guiding decision-making. This simplicity and clarity make GRMobiNet a robust tool for image classification, suitable for integration into various applications that require reliable and confident predictions.

### *3.3. GRMobiNet Model Construction*

The construction of GRMobiNet follows a systematic approach to develop an efficient image classification model optimized for deployment on RCE devices. The process involves several key steps:

#### *Step 1: Setting the Baseline Model as MobileNet*

The foundation of the GRMobiNet architecture is the well-established MobileNet model, known for its efficiency and high performance in image classification tasks. As explained before, the MobileNet includes Depthwise Separable Convolutions technique which reduces the number of parameters and computational cost by factorizing a standard convolution into a depthwise convolution followed by a pointwise convolution. Moreover, the baseline model contains initial convolution layers which consist of a 3x3 convolution with a stride of 2, followed by batch normalization and

ReLU activation functions. These layers are crucial for initial feature extraction and reducing the spatial dimensions of the input image in the GRMobiNet.

#### *Step 2: Applying Optimization Techniques- Compound Scaling and GAP*

To enhance the performance of the baseline model, two key optimization techniques are added to the baseline model. Firstly, CompScale which technique involves scaling the depth, width, and resolution of the model uniformly based on predefined multipliers. For GRMobiNet, the depth and width multiplier ( $\alpha$ ) is set to 1.0, and the resolution multiplier ( $\phi$ ) is set to 1.2. Moreover, the initial input resolution of 224x224 is scaled to 268x268, increasing the resolution while maintaining the overall efficiency of the model. This scaling enhances the model's capacity to capture finer details in the images, leading to improved accuracy. Secondly, the GAP, which replaces the fully connected layers typically found in conventional neural networks. By averaging the spatial dimensions of the feature maps, GAP reduces the risk of overfitting and decreases the number of parameters, further enhancing the model's efficiency.

#### *Step 3: Fine-Tuning on ImageNet-10 Dataset*

The fine-tuning process of the GRMobiNet architecture involves several key steps. First, the last layer of the original MobileNet model is removed to adapt it for 10-class classification, replacing the original output layer. This removed layer is then replaced with a new Dense layer consisting of 10 units, each corresponding to a class in the ImageNet-10 dataset, and using a softmax activation function to output probabilities for each class. The model's architecture is then adjusted to incorporate this new output layer, with settings of `num_classes=10` and `activation='softmax'` to ensure effective handling of the multi-class classification task. Finally, a new model is constructed by combining the modified architecture with the new output layer, resulting in the GRMobiNet model, which integrates the original MobileNet framework with the updated output configuration for 10-class classification.

The resulting GRMobiNet model consists of 89 layers and 3,239,114 parameters, with a model size of 12.60 MB. A part of constructed GRMobiNet model architecture is shown in Figure 3. The model begins with an input layer that accepts images of size 268x268 pixels with 3 color channels (RGB). The initial convolution layer reduces the spatial dimensions of the input, followed by depthwise separable convolution layers that minimize computational cost. These layers consist of depthwise convolutions, where each input channel is filtered separately, followed by pointwise convolutions to combine the filtered channels. Downsampling blocks, using zero-padding and depthwise convolutions, reduce spatial dimensions to abstract higher-level features. The network alternates between depthwise separable convolutions and ReLU activations to build complex feature maps. A Global Average Pooling layer condenses the feature maps into a single 1024-dimensional vector, which is then passed to the final dense classification layer, producing 10 outputs for the 10-class classification task. This architecture strikes a balance between computational efficiency and high performance, making it well-suited for RCEs.

### *3.4. Model Training*

The GRMobiNet architecture, after fine-tuning (as stated in Section 3.3), was trained using the specified hyperparameters listed in Table 4, which includes essential parameters such as the input shape, batch size, learning rate, optimizer, loss function, and the number of epochs. Additionally, the software (SW) and hardware (HW) configurations used for training are detailed in Table 5, outlining the computational environment that supported the training process. This environment played a critical role in optimizing the model's efficiency, ensuring that it could leverage available resources to achieve optimal performance. The training process itself focused on optimizing the model's parameters by utilizing the ImageNet-10 dataset, which contains 10 distinct image categories. The model learns to classify these images accurately by adjusting its weights iteratively based on the input data. During each epoch, the model processes the training data, calculates the loss using the loss function, and adjusts the weights accordingly to minimize this loss. Over multiple epochs, this process enables the model to improve its generalization capability, gradually refining its performance and increasing accuracy. This iterative adjustment of weights helps the model learn the underlying patterns and features in the data, ultimately leading to a well-trained model capable of effectively classifying images into the correct categories.

The goal of the training process was to enable the model to accurately identify and classify images into image class categories. To optimize the model's learning, the Adam optimizer was employed due to its efficiency in handling large datasets and its ability to adaptively adjust the learning rate during training. This optimizer helped ensure that the model converged more quickly and effectively. The training process used a categorical cross-entropy loss function, which is commonly used for multi-class classification problems. This loss function measures the difference between the model's predicted probabilities and the actual labels of the images, guiding the model in adjusting its weights to reduce the prediction errors. To evaluate the performance of the model throughout the training, accuracy was chosen as the validation metric, providing a straightforward measurement of how many images were correctly classified by the model during each training epoch.

The model was trained over 50 epochs, meaning that it went through the entire training dataset 50 times, with the weights being updated after each pass through the data. A learning rate of 0.0001 was used, which is relatively small, ensuring that the model's weights were adjusted gradually to prevent exceeding the optimal solution. The combination of these hyperparameters as summarized in Table 4—Adam optimizer, categorical cross-entropy loss function, accuracy

metric, a learning rate of 0.0001, and 50 epochs—allowed the model to progressively improve its classification accuracy while minimizing errors, resulting in a well-tuned model capable of handling the 10-class classification task.

| Layer (type)                        | Output Shape          | Param # |
|-------------------------------------|-----------------------|---------|
| input (InputLayer)                  | [(None, 268, 268, 3)] | 0       |
| conv1_pad (ZeroPadding2D)           | (None, 269, 269, 3)   | 0       |
| conv1 (Conv2D)                      | (None, 134, 134, 32)  | 864     |
| conv1_bn (BatchNormalization)       | (None, 134, 134, 32)  | 128     |
| conv1_relu (ReLU)                   | (None, 134, 134, 32)  | 0       |
| conv_dw_1 (DepthwiseConv2D)         | (None, 134, 134, 32)  | 288     |
| conv_dw_1_bn (BatchNormaliza        | (None, 134, 134, 32)  | 128     |
| conv_dw_1_relu (ReLU)               | (None, 134, 134, 32)  | 0       |
| *****                               |                       |         |
| conv_dw_13 (DepthwiseConv2D)        | (None, 8, 8, 1024)    | 9216    |
| conv_dw_13_bn (BatchNormaliz        | (None, 8, 8, 1024)    | 4096    |
| conv_dw_13_relu (ReLU)              | (None, 8, 8, 1024)    | 0       |
| conv_pw_13 (Conv2D)                 | (None, 8, 8, 1024)    | 1048576 |
| conv_pw_13_bn (BatchNormaliz        | (None, 8, 8, 1024)    | 4096    |
| conv_pw_13_relu (ReLU)              | (None, 8, 8, 1024)    | 0       |
| global_average_pooling2d_3          | (None, 1024)          | 0       |
| dense_2 (Dense)                     | (None, 10)            | 10250   |
| Number of layers: 89                |                       |         |
| Total number of parameters: 3239114 |                       |         |
| Model size: 12.60 MB                |                       |         |

Fig.3. Architecture of GRMobiNet model

Table 4. Hyperparameter configuration of GRMobiNet

| No. | Hyperparameters                         | Value                    | Justification                                                                                                                                        |
|-----|-----------------------------------------|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1.  | Input shape                             | 268X268X3                | Higher than the default 224×224 to enhance feature granularity for fine-grained image classification; Aligns with compound scaling ( $\phi = 1.2$ ). |
| 2.  | Batch size                              | 16                       | Balanced memory usage and training hardware; avoids overloading or slow learning. (empirical value)                                                  |
| 3.  | Number of image class                   | 10                       | Selected from ImageNet10 for early-stage prototyping.                                                                                                |
| 4.  | Compound scaling- width multiplier      | 1.0.                     | Maintains original channel depth, ensuring computational feasibility while preserving representational power. (empirical)                            |
| 5.  | Compound scaling- resolution multiplier | 1.2                      | Slightly increases input resolution to improve accuracy without a large increase in parameter count. (empirical)                                     |
| 6.  | Activation function of input layer      | ReLU                     | Chosen for its simplicity, computational efficiency, and effectiveness in preventing vanishing gradients.                                            |
| 7.  | Activation function of output layer     | softmax                  | Produces a probability distribution over 10 classes for classification; standard for categorical output and multi-class classification;              |
| 8.  | Number of stride                        | 2                        | Enables effective down sampling of feature maps, reducing spatial dimensions and improving compute efficiency.                                       |
| 9.  | Optimizer                               | adam                     | Adaptive learning rate and efficient convergence, especially effective on small datasets and lightweight models.                                     |
| 10. | Loss function                           | categorical_crossentropy | Standard for multi-class classification                                                                                                              |
| 11. | Metric                                  | accuracy                 | Measures classification precision; aligns with study goal of maximizing accuracy on RCEs.                                                            |
| 12. | Learning rate                           | 0.0001                   | Provide stable convergence during experimentation                                                                                                    |
| 13. | Number of Epochs:                       | 50                       | Allowed sufficient learning without overfitting (empirical)                                                                                          |

Due to the variability in results based on the selected variables, identifying suitable parameters for the classification task was initially challenging. Following a trial-and-error approach, as suggested by Passos and Mishra (2022) and Kamal and Hamid (2023), the study explored various parameter combinations through extensive experimentation to determine the optimal configuration. The finalized hyperparameters are summarized in Table XX

Due to the variability in results based on the selected variables, identifying suitable parameters for the classification task was initially challenging. Following a trial-and-error approach [49, 50], the study explored various parameter combinations through extensive experimentation to determine the optimal configuration as given in Table 4.

Table 5. Software and hardware configurations

| Software                                                                                                                                        | Hardware                                                       | Used for                |
|-------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------|-------------------------|
| Windows 10<br>Python 3.8.19,<br>Tensor Flow 2.3.0,<br>NumPy 1.18.5,<br>Matplotlib 3.4.3,<br>Pandas 1.2.4<br>Keras 2.4.0<br>Sklearn-pandas 2.2.0 | Intel i7-3770 CPU @ 3.5GHz (8 CPUs) with 12 GB of RAM          | Training the model      |
|                                                                                                                                                 | Intel Celeron CPU N2920 @1.86 GHz (4 CPU) With and 4 GB of RAM | Experimenting the model |

### 3.5. Dataset

In this research, the ImageNet-10 dataset, a subset of the widely used ImageNet-1K dataset [51-54], was used for training, validating, and testing the GRMobiNet model. ImageNet-10 was designed to address challenges associated with large-scale datasets like ImageNet-1K by focusing on a smaller selection of 10 categories, as representative of 1000 categories, which helps reduce training time and allows for quicker construction and validation of new models. Figure 4 illustrates the relationship between ImageNet-1K and ImageNet-10, highlighting how the latter is derived by selecting a subset of 10 classes from the original 1,000, thus creating a more focused dataset for image classification tasks.

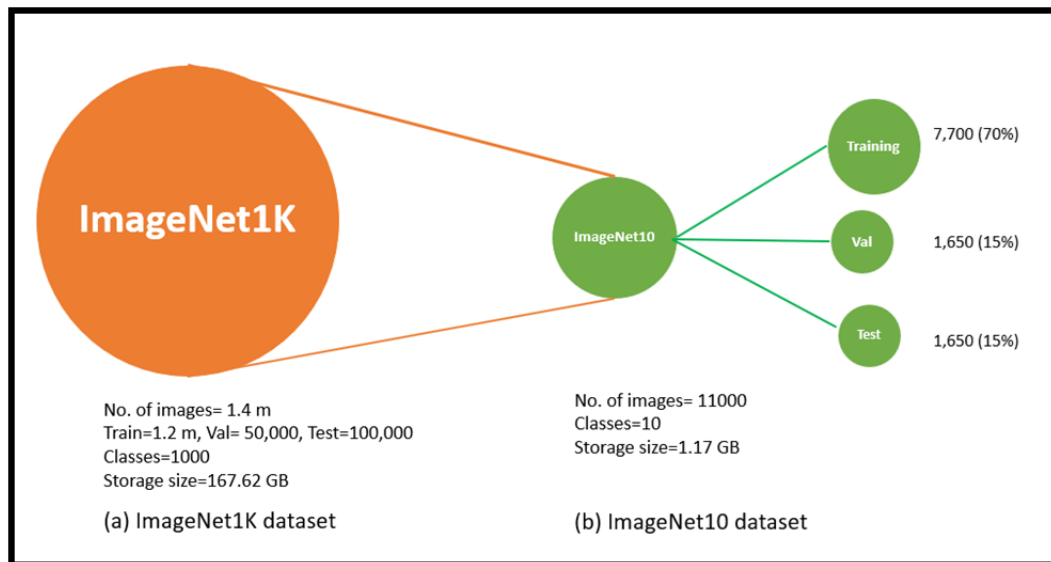


Fig.4. Outline for setting of ImageNet-10 from ImageNet-1K

Table 6. Image categories of ImageNet-10 dataset with class ID and class name

| Class ID | Class Name |
|----------|------------|
| 1        | aeroplane  |
| 2        | balloon    |
| 3        | cock       |
| 4        | dog        |
| 5        | goldfish   |
| 6        | horse      |
| 7        | lion       |
| 8        | panda      |
| 9        | parrot     |
| 10       | shark      |

The ImageNet-10 dataset comprises images from ten distinct categories, meticulously organized to provide a thorough assessment of models' performance. The selected categories include aeroplane, balloon, cock, dog, goldfish, horse, lion, panda, parrot and shark as shown in Table 6. This diverse selection ensures the dataset covers a broad spectrum of object types, from animals and fish to everyday items, thereby simulating real-world conditions where models must accurately identify and classify various objects.

By employing the ImageNet-10 dataset, the study aims to streamline the development cycle and enhance the efficiency of the model training process. The diverse selection of images tests the models' ability to accurately classify a wide range of objects, providing valuable insights into their generalization capabilities. This approach ensures a robust and comprehensive evaluation, enabling quick iteration and validation of new models.

The ImageNet-10 dataset is organized into three distinct subsets to facilitate effective model training, validation, and testing, as shown in Figure 5. The distribution of the 11,000 total images is as follows aligned with [55]:

- Training Set: Comprising 7,700 images, which accounts for 70% of the total dataset. This set is used to train the model, allowing it to learn and recognize patterns within the data.
- Validation Set: Consisting of 1,650 images, representing 15% of the total dataset. The validation set is used during the training process to tune the model's parameters and to monitor its performance, helping to prevent overfitting.
- Test Set: Also made up of 1,650 images, equating to 15% of the total dataset. The test set is utilized to evaluate the final performance of the trained model, providing an unbiased assessment of its generalization capabilities on unseen data.

This structured distribution ensures a comprehensive approach to model development, allowing for thorough training, validation, and evaluation.

### 3.6. Justification for the Selection of ImageNet10

The selection of ImageNet10 as the training dataset for GRMobiNet is grounded in several key factors that ensure its representativeness of real-world applications, as well as its sufficiency for evaluating the model's performance in resource-constrained environments (RCEs). While ImageNet10 is a subset of the larger ImageNet-1K dataset, it strikes a balance between diversity, complexity, and manageability, making it an appropriate choice for initial model evaluation.

#### a. Representativeness of Real-World Applications

ImageNet10 includes a range of visually diverse categories such as animals, vehicles, and natural scenes. These categories are highly relevant to a wide array of real-world applications, including autonomous systems, medical image analysis, wildlife monitoring, and plant disease detection. For instance, the inclusion of animals like lions and dogs, as well as everyday objects like aeroplanes and sharks, ensures that the model can generalize across multiple domains—both for general-purpose image classification and domain-specific tasks like medical imaging and plant disease detection. This diversity supports the development of a model that is applicable to various real-world scenarios, ensuring that it can be adapted to future, more specialized tasks.

#### b. Diversity and Complexity

Although ImageNet10 is a subset of the larger ImageNet dataset, it offers a high degree of diversity in terms of object types and visual characteristics. This allows GRMobiNet to develop a rich set of feature representations that are applicable to a range of domains. In contrast to simpler datasets like MNIST, which focus on small grayscale digits, ImageNet10 includes high-resolution color images that represent more complex, realistic tasks. This characteristic of the dataset provides a strong foundation for building a model that can handle varied and challenging visual data. Consequently, training on ImageNet10 ensures that GRMobiNet learns more robust feature extraction techniques, making it adaptable to a wider range of applications in real-world scenarios.

#### c. Sufficiency for Initial Evaluation

While ImageNet10 does not capture every possible real-world use case, it is sufficiently diverse and complex to test the core functionalities of GRMobiNet. The dataset's moderate number of classes (10) enables efficient experimentation and faster iterations, making it an ideal choice for early-stage model development. It provides enough variety and complexity to evaluate key aspects such as classification accuracy, feature extraction, and model efficiency, while remaining manageable for rapid testing. This ensures that the model can undergo thorough initial evaluations before progressing to more complex, specialized datasets that address specific domains such as medical or plant disease detection.

### 3.6. Evaluation Metrics

Evaluation metrics can show how the performance of the GRMobiNet model on image classification in RCE device. The commonly used evolution metrics for measure performance such as prediction accuracy, precision, recall

and F1-score [45-57] were used for evaluation. When analyzing a DNN model for image classification, it is essential to take into account all of these metrics together to gain a comprehensive understanding of the model's performance, which is crucial for addressing the research objectives of this paper. The mathematical formulas to define the above metrics are as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

$$F1-Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

Where True Positives (TP) refer to instances where the GRMobiNet model correctly classifies an image as belonging to a specific class. True Negatives (TN) are instances where the model accurately identifies an image as not belonging to a particular class. False Positives (FP) occur when the model incorrectly classifies an image as belonging to a class, even though it does not, essentially labeling a negative instance as positive. False Negatives (FN) represent cases where the model wrongly classifies an image as not belonging to a class when it actually does, failing to recognize a positive instance.

```
Epoch 1/50
481/481 [=====] - 1723s 4s/step - loss: 1.7034 - accuracy: 0.4056 - val_loss: 6.3110 - val_accuracy: 0.0996
Epoch 2/50
481/481 [=====] - 1703s 4s/step - loss: 1.2336 - accuracy: 0.5765 - val_loss: 2.4605 - val_accuracy: 0.4840
*****
Epoch 9/50
481/481 [=====] - 1698s 4s/step - loss: 0.6690 - accuracy: 0.7866 - val_loss: 1.2164 - val_accuracy: 0.6874
Epoch 10/50
481/481 [=====] - 1699s 4s/step - loss: 0.6362 - accuracy: 0.7954 - val_loss: 0.9065 - val_accuracy: 0.7170
Epoch 11/50
481/481 [=====] - 1708s 4s/step - loss: 0.6026 - accuracy: 0.8122 - val_loss: 1.0359 - val_accuracy: 0.7043
*****
Epoch 23/50
481/481 [=====] - 1697s 4s/step - loss: 0.3354 - accuracy: 0.8920 - val_loss: 0.5715 - val_accuracy: 0.8358
Epoch 24/50
481/481 [=====] - 1696s 4s/step - loss: 0.3055 - accuracy: 0.9012 - val_loss: 0.6403 - val_accuracy: 0.8383
Epoch 25/50
481/481 [=====] - 1697s 4s/step - loss: 0.3016 - accuracy: 0.9028 - val_loss: 0.5907 - val_accuracy: 0.8286
Epoch 26/50
481/481 [=====] - 1699s 4s/step - loss: 0.3100 - accuracy: 0.8998 - val_loss: 0.6113 - val_accuracy: 0.8401
*****
481/481 [=====] - 1699s 4s/step - loss: 0.2550 - accuracy: 0.9167 - val_loss: 0.3999 - val_accuracy: 0.8787
Epoch 31/50
481/481 [=====] - 1696s 4s/step - loss: 0.2471 - accuracy: 0.9219 - val_loss: 0.4560 - val_accuracy: 0.8721
Epoch 32/50
481/481 [=====] - 1706s 4s/step - loss: 0.2441 - accuracy: 0.9218 - val_loss: 0.4577 - val_accuracy: 0.8727
Epoch 33/50
*****
Epoch 39/50
481/481 [=====] - 1691s 4s/step - loss: 0.1936 - accuracy: 0.9377 - val_loss: 0.5484 - val_accuracy: 0.8624
Epoch 40/50
481/481 [=====] - 1699s 4s/step - loss: 0.1700 - accuracy: 0.9442 - val_loss: 0.3721 - val_accuracy: 0.8859
*****
Epoch 48/50
481/481 [=====] - 1693s 4s/step - loss: 0.1456 - accuracy: 0.9545 - val_loss: 0.2986 - val_accuracy: 0.9161
Epoch 49/50
481/481 [=====] - 1697s 4s/step - loss: 0.1393 - accuracy: 0.9529 - val_loss: 0.5257 - val_accuracy: 0.8636
Epoch 50/50
481/481 [=====] - 1691s 4s/step - loss: 0.1382 - accuracy: 0.9543 - val_loss: 0.3660 - val_accuracy: 0.9176
```

Fig.5. A part of GRMobiNet training result

## 4. Result and Discussion

### 4.1. Training Result

The GRMobiNet model was trained using the ImageNet-10 training dataset on a high-performance hardware device, including an Intel i7-3770 CPU @ 3.5GHz (8 CPUs) and 12 GB of RAM, with the software configuration outlined in Table 4. The following Figure 5 shows a part of training and validation result including training loss and training accuracy for the training dataset and, validation loss and validation accuracy for the testing dataset for the

selected epoch 50 model training. The Figure 6 shows the training curve [57] obtained from the GRMobiNet training time. The GRMobiNet model was trained for 50 epochs, showing steady improvements in both training and validation accuracy. Initially, at Epoch 1, the model had a training accuracy of 40.56% and a validation accuracy of 9.96%. By Epoch 10, the training accuracy rose to 79.54%, and validation accuracy reached 71.70%. At Epoch 25, the training accuracy improved to 90.28%, with validation accuracy stabilizing at 83.86%. By Epoch 50, the model achieved a training accuracy of 95.43% and a validation accuracy of 91.76%, with a validation loss of 0.3660. These results show strong performance, indicating effective learning with no significant overfitting.

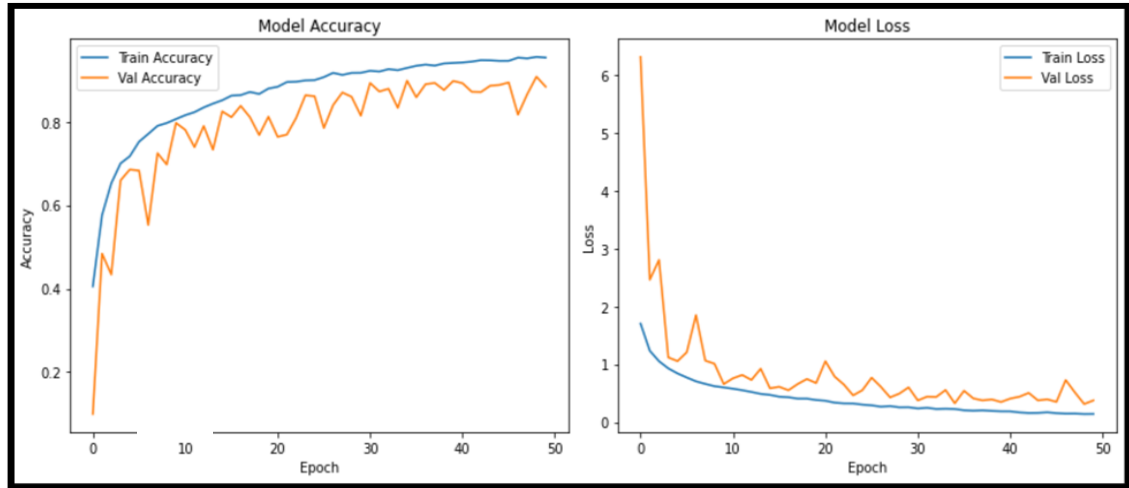


Fig.6. Model Training curve; (a) shows the changes of training and validation accuracies with epoch; (b) shows changes of training loss and validation loss with epoch

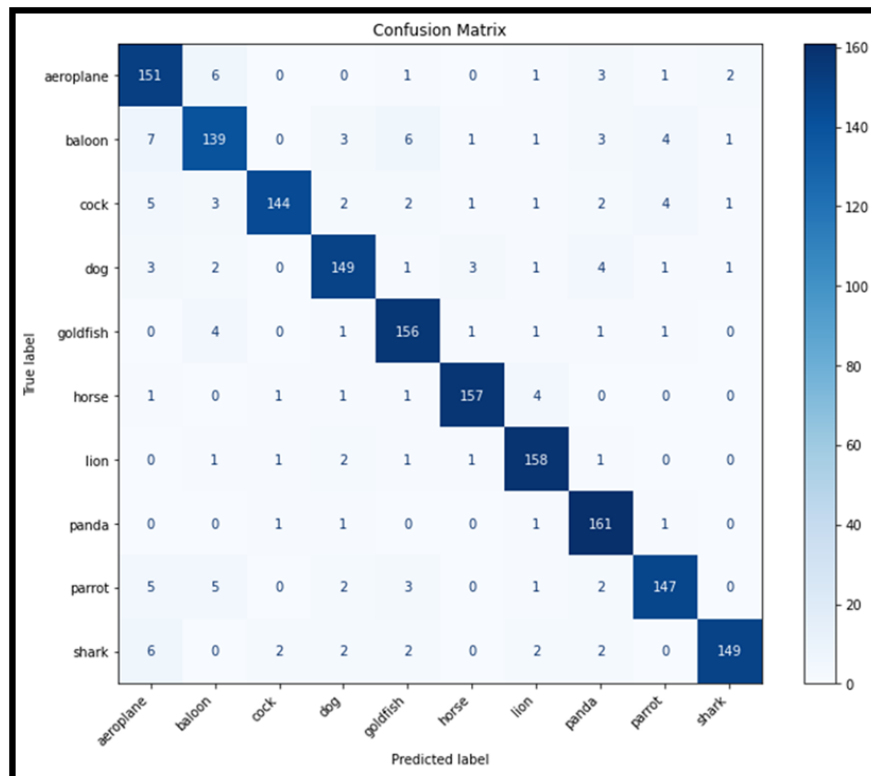


Fig.7. Confusion matrix of GRMobiNet for the ImageNet-10 testing dataset

#### 4.2. Testing Result

To evaluate the effectiveness of the GRMobiNet model, a comprehensive experiment was conducted using the testing dataset of ImageNet-10. This testing dataset, which consists of 1650 unseen images across 10 distinct categories, was specifically selected to measure the model's generalization capabilities and accuracy in a real-world image classification scenario. The experiment was carried out on a RCE setting with an Intel Celeron CPU N2920 @ 1.86

GHz (4 CPUs) and 4 GB of RAM. While this hardware may seem outdated by contemporary standards, it was intentionally selected to reflect the constraints faced by many RCE devices, including legacy embedded systems, budget mobile devices, and IoT devices. These devices often have limited computational resources and memory, underscoring the importance of optimizing DNN models for efficient deployment in such environments.

The results of the experiment were captured and presented in a confusion matrix [57] to visualize the classification result, as shown in Figure 7. This matrix provided detailed insights into the classification performance of the model across the 10 categories, allowing for an in-depth analysis of its accuracy, precision, recall, and F1-scores.

In Figure 7 confusion matrix, the diagonal elements indicate high values, reflecting the GRMobiNet model's strong performance in accurately predicting most classes. Specifically, the model successfully classified 97.57% of Panda instances (161/165), 95.77% of Lion instances (158/165), and 95.15% of Horse instances (157/165), demonstrating excellent accuracy. Additionally, Goldfish (94.55% – 156/165), Dog (90.30% – 149/165), and Cock (87.27% – 144/165) also exhibited solid performance, with only minor misclassifications into closely related categories. To further assess the model's effectiveness, key metrics such as accuracy, precision, recall, F1-score, parameter number, and model size were computed and tabulated in Table 7.

Table 7. Image categories of ImageNet-10 dataset with class ID and class name

| Model     | Accuracy | Precision | Recall | F1-score | Param (M) | Model Size (MB) |
|-----------|----------|-----------|--------|----------|-----------|-----------------|
| GRMobiNet | 92       | 92        | 92     | 92       | 3.2       | 12.6            |

The analysis presented in Table 7 demonstrates that the GRMobiNet model performs exceptionally well across multiple evaluation metrics. It achieves an accuracy of 92%, meaning it correctly classifies 92% of all instances in the dataset. The model also shows high precision and recall, both at 92%, which indicates its effectiveness in accurately identifying instances of the positive class while minimizing false positives and false negatives. The F1-score, a combination of precision and recall, is also 92%, reflecting balanced performance. With 3.2 million parameters, the model maintains a moderate level of complexity and requires only 12.6 MB of memory for storage, ensuring efficient memory usage. These results align closely with the research objectives of reducing computational complexity to fewer than 4 million parameters, minimizing memory usage to under 16 MB, and achieving an accuracy above 80%.

#### 4.3. Model Comparison Result

In addition to the performance analysis mentioned above, this paper presents a comprehensive comparative analysis to evaluate the efficiency and effectiveness of the GRMobiNet model in comparison to other advanced model [57]. The analysis focuses on key performance metrics, including inference accuracy, CPU usage time, model size, and the number of parameters. GRMobiNet's performance is compared with four state-of-the-art models: MobileNetV2 [46], EfficientNetB0 [14], EfficientNetB1[22], and ResNet50 [25] (Lawrence and Zhang 2019, Al-Jubouri and Mahmmoud 2021). To ensure a fair comparison, all models were fine-tuned and re-trained using the ImageNet-10 dataset.

The comparative study was conducted in the same resource-constrained environment (Table 4), using a laptop equipped with an Intel Celeron CPU N2920 @ 1.86 GHz (4 CPUs) and 4 GB of RAM. This setup simulates the hardware limitations typical of real-world mobile devices and embedded systems. Each model, including GRMobiNet, was deployed on this laptop to evaluate its performance in image classification tasks under these conditions. While the models being compared may differ in their computational complexity and hardware requirements, we took several steps to ensure fairness and consistency in the evaluation. All models were run on the same laptop to eliminate any hardware variations, ensuring that performance differences were due to the models themselves and not hardware discrepancies. To address the potential impact of resource-intensive models, we monitored CPU usage and memory consumption throughout the evaluation, ensuring no model was unfairly advantaged due to its computational needs. Additionally, all models, including GRMobiNet, were fine-tuned and trained using the same ImageNet-10 dataset and identical preprocessing steps, including data augmentation, image resizing, and normalization, to avoid any unfair advantages in data preparation. We also recorded inference speed and CPU usage for each model, ensuring that the comparisons were not skewed by hardware limitations. By maintaining these controlled conditions, we ensured that all models were evaluated fairly, and the results accurately reflect each model's true performance under resource-constrained environments.

For this evaluation, a dataset of 100 unseen real-world images (an external dataset) was used, organized into 10 categories of images with 10 images per category. The images were stored on the laptop, and each model was run separately to classify them. This approach allows for a direct comparison of model performance under identical conditions. The main objective of the experiment was to assess how well GRMobiNet and other models perform in accurately classifying real-world images. To ensure a fair comparison, all models were evaluated under the above same conditions. Key performance metrics—including inference accuracy, inference speed, model size, and the number of parameters—were analyzed to compare GRMobiNet with other advanced DNN models. Inference accuracy for each model in the real-world image classification task was calculated using their confusion matrices, as shown in Figure 8. Additionally, other metrics, including the number of parameters, model size, and inference time, were recorded and are summarized in Table 8.

The experimental results reveal that GRMobiNet outperforms all the other models across multiple areas. In terms of inference accuracy, GRMobiNet achieved an impressive 80%, surpassing MobileNetV2 (57%), EfficientNet-B0 (17%), EfficientNet-B1 (34%), and ResNet-50 (46%). This demonstrates GRMobiNet's superior ability to accurately classify real-world images. Regarding inference speed, GRMobiNet exhibited a mean inference time of 959.14 milliseconds, which is faster than MobileNetV2 (1081.16 ms), EfficientNet-B0 (1170.47 ms), EfficientNet-B1 (1471.49 ms), and ResNet-50 (6122.22 ms). This faster speed makes GRMobiNet a more efficient choice for real-time applications. Furthermore, GRMobiNet excels in model size and the number of parameters. It utilizes only 3.2 million parameters, which is fewer than MobileNetV2 (3.5M), EfficientNet-B0 (5.3M), EfficientNet-B1 (7.9M), and ResNet-50 (25.6M). Additionally, its compact model size of 12.6 MB is significantly smaller than the other models: EfficientNet-B0 (29 MB), EfficientNet-B1 (31 MB), and ResNet-50 (98 MB). This smaller model size and fewer parameters make GRMobiNet ideal for deployment in RCEs where memory and storage are limited.

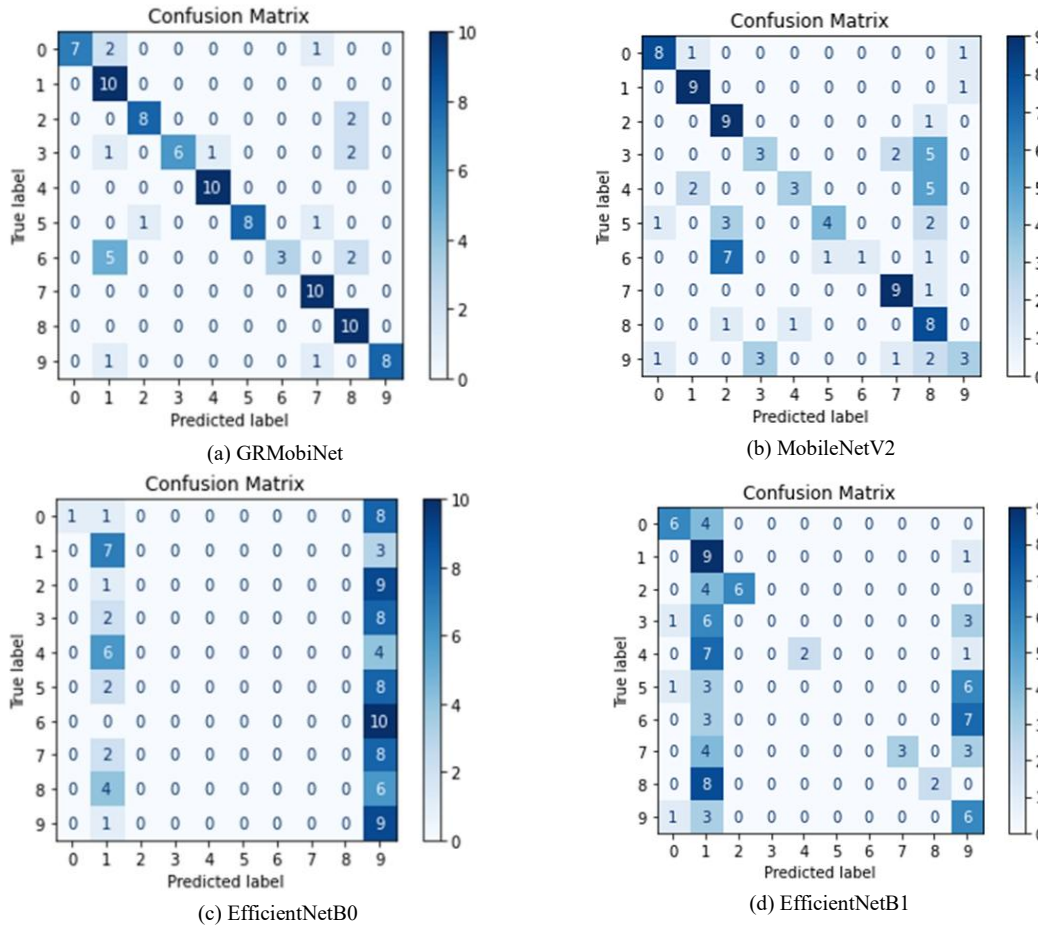
Table 8. Comparison of computational complexity of GRMobiNet with other advanced DNN models

| Metrics            | GRMobiNet | MobileNetV2 | EfficientNet-B0 | EfficientNet-B1 | ResNet-50 |
|--------------------|-----------|-------------|-----------------|-----------------|-----------|
| Accuracy (%)       | 80        | 57          | 17              | 34              | 46        |
| Mean Time (ms)     | 959.14    | 1081.16     | 1170.47         | 1471.49         | 6122.22   |
| Parameters No. (M) | 3.2       | 3.5         | 5.3             | 7.9             | 25.6      |
| Model size (MB)    | 12.6      | 14          | 29              | 31              | 98        |

#### 4.4. Improvement of GRMobiNet over MobileNet

The GRMobiNet model builds upon the MobileNet architecture by incorporating targeted architectural enhancements aimed at improving performance in RCEs. Although MobileNet is already a highly optimized lightweight DNN, GRMobiNet surpasses it by delivering significant gains in accuracy, inference speed, model size, and parameter efficiency, without requiring specialized hardware or external optimizations.

Table 9 presents a comparative analysis of GRMobiNet against MobileNetV2 (advanced version of MobileNet) across four key metrics extracted from Table 8.



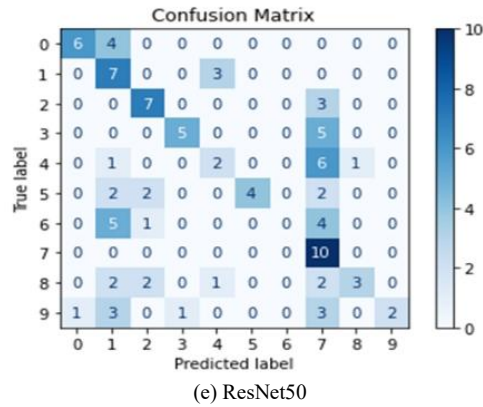


Fig.8. Confusion matrixes of the experimental result of (a) GRMobiNet, (b) MobileNetV2, (c) EfficientNetB0, (d) EfficientNetB1, and (e) ResNet50

Table 9. Comparative Evaluation of GRMobiNet and MobileNetV2

| DNN Model     | Accuracy | Time (ms) for inference step | Model Size | Parameters |
|---------------|----------|------------------------------|------------|------------|
| MobileNetV2   | 57       | 1081.16                      | 14         | 3.5        |
| GRMobiNet     | 80       | 959.14                       | 12.6       | 3.2        |
| % improvement | 23       | 11.29                        | 10         | 8.6        |

- **Accuracy Improvement (+23%):** GRMobiNet achieves a 23% improvement in classification accuracy over MobileNetV2 (80% vs. 57%). This substantial gain underscores the effectiveness of GRMobiNet's design, which incorporates features such as compound scaling, efficient feature reuse, and global average pooling. These enhancements enable the model to extract and generalize image features more effectively. This accuracy boost is especially critical in high-impact domains such as medical diagnostics, agricultural disease detection, and autonomous systems, where predictive precision is paramount.
- **Inference Time Reduction (−11.29%):** GRMobiNet reduces average inference time from 1081.16 ms to 959.14 ms, reflecting an 11.29% speed-up. This is particularly important for real-time and interactive applications, where lower latency improves user experience and system responsiveness. Notably, this gain is achieved through intrinsic architectural efficiency, rather than hardware-level acceleration, which highlights the practicality of GRMobiNet in low-resource deployment settings.
- **Model Size Reduction (−10%):** With a model size of 12.6 MB (vs. 14 MB for MobileNetV2), GRMobiNet is more suitable for storage-limited environments such as smartphones, IoT devices, and embedded systems. The size reduction is achieved through quantization and the replacement of fully connected layers with global average pooling. These optimizations enable on-device inference, supporting applications that require offline functionality, low latency, or enhanced privacy.
- **Parameter Count Reduction (−8.6%):** GRMobiNet contains 3.2 million parameters, a reduction of 8.6% from MobileNetV2's 3.5 million. This lower parameter count translates to reduced memory usage, lower power consumption, and faster inference, making the model more viable for battery-operated or thermally sensitive devices. Crucially, this reduction is achieved without sacrificing accuracy, underscoring the model's optimization efficiency.

## 5. Conclusions

This research introduces GRMobiNet, an optimized DNN model specifically designed for efficient image classification on RCE devices. The model addresses challenges related to reducing computational complexity, minimizing memory usage, and maximizing accuracy, making it well-suited for deployment in such environments.

To achieve a balance between computational complexity and model accuracy, GRMobiNet incorporates a set of targeted architectural optimizations that reduce resource demands while maintaining high predictive performance. Specifically, GRMobiNet integrates depthwise separable convolutions to reduce parameter count and computation, compound scaling to proportionally adjust model depth, width, and resolution, and global average pooling to replace dense fully connected layers, thereby lowering model complexity and improving generalization. Additionally, quantization is applied to reduce precision in weights and activations, resulting in faster inference and smaller memory requirements with minimal accuracy degradation. These techniques, detailed in Table 3, enable GRMobiNet to achieve a compact model with only 3.2 million parameters and a memory footprint of 12.6 MB, while delivering a classification accuracy of 92% on the ImageNet-10 dataset. Moreover, in real-world testing on 100 unseen images, GRMobiNet achieved the highest inference accuracy (80%) among four well-known models, demonstrating that the chosen trade-

offs effectively optimize efficiency without compromising accuracy.

Notably, GRMobiNet delivers these gains with 11.29% faster inference (compared to MobileNetV2), 10% reduction in model size, 8.6% fewer parameters. These improvements result from several key architectural innovations, including compound scaling, depthwise separable grouped convolutions, global average pooling, and 8-bit quantization design, all contributing to enhanced generalization and operational efficiency.

The use of 8-bit quantization in GRMobiNet was primarily aimed at reducing model size and accelerating inference, especially for deployment on memory- and compute-constrained devices. While quantization can introduce minor reductions in accuracy due to the loss of numerical precision, in our experiments, GRMobiNet maintained high classification performance after quantization. Furthermore, quantization significantly reduced the model's memory footprint and improved inference speed, which is critical for real-time applications. These findings suggest that the trade-offs introduced by 8-bit quantization are acceptable and well-compensated by the performance gains in resource-constrained scenarios.

GRMobiNet represents a significant advancement in the development of resource-efficient DNN models, offering a balanced trade-off between high accuracy and low computational complexity. Its success in achieving the research objectives positions it as a promising candidate for real-world deployment in RCE applications.

The experimental results, using the ImageNet10 testing dataset, show that GRMobiNet successfully achieves high accuracy and reduced computational complexity compared to several popular models, including MobileNetV2, EfficientNet-B0, and ResNet50. However, while the current paper highlights the strengths of GRMobiNet against these models, it acknowledges the need for a more in-depth comparison with recent lightweight models and models specifically designed for RCEs.

In future work, we will expand the comparative analysis to include newer models such as ShuffleNetV2[58], GhostNet [59], MobileNetV3 [17] and YOLOv4 [60] which have been specifically designed to improve performance while maintaining low computational overhead. Additionally, we will conduct tests on a broader set of datasets, including more real-world images that better reflect the diversity of applications in resource-constrained environments. These future experiments will also consider real-world use cases to assess the practical deployment of GRMobiNet in diverse environments, such as mobile health applications, autonomous systems, and IoT devices.

Although GRMobiNet's model size is reported as 12.6 MB, we further examined its deployability in memory-limited environments by running experiments on the low-spec laptop equipped with 4 GB RAM and an Intel Celeron CPU (Table 5), which simulates resource-constrained hardware. The model loaded and executed successfully, without memory overflows or runtime delays, demonstrating its suitability for low-memory environments. However, we acknowledge that real-world RCE devices, such as Raspberry Pi boards or microcontrollers, may present different memory constraints. In future work, we plan to deploy GRMobiNet on a wider range of edge platforms to evaluate actual memory consumption and runtime performance more comprehensively. This will allow us to validate its efficiency across diverse deployment scenarios and ensure compatibility with more constrained hardware environments.

By exploring a broader range of models and datasets, as well as evaluating the model's performance in real-world scenarios, we aim to provide a more comprehensive understanding of GRMobiNet's performance and its relative advantages over more recent advancements in lightweight deep learning models. This will help solidify GRMobiNet's position as an optimized solution for resource-constrained applications.

### Author Contributions Statement

Raafi Careem - Design, development, model training and validation, deployment, and experimentation of the GRMobiNet model, and writing the manuscript.

Md Gapar Md Johar - supervised the project, provided comments and feedback, checked the experimental results, reviewed and edited the manuscript, checked the clarity and coherence.

All authors have read and agreed to the published version of the manuscript.

### Conflict of Interest Statement

The authors declare no conflicts of interest

### Funding Declaration

No funding received for this work.

### Data Availability Statement

This study analyzed publicly available dataset. The dataset can be found in <https://www.kaggle.com> accessed on 22<sup>nd</sup> March 2026.

### Ethical Declaration

This research did not require ethical approval, as it involved no human participants, animal subjects, or experimental procedures requiring ethical review.

## Acknowledgments

We sincerely thank the experts for their professional evaluation and valuable recommendations, which have contributed to the quality of the experiment and the reliability of its results

## Declaration of Generative AI in Scholarly Writing

The manuscript was prepared with limited assistance from Generative AI tools (ChatGPT and QuilBot) solely for language editing and clarity enhancement. All scientific content, analyses, interpretations, and conclusions presented in this work were authored, verified, and are the full responsibility of the authors.

## Abbreviations

The following abbreviations are used in this manuscript:

DNN- Deep Neural Networks  
 RCE- Resource Constrained Environment  
 DSConv - depthwise separable convolution  
 ComSc - compound scaling  
 GAP - global average pooling

## References

- [1] A. Bhargava and A. Bansal, "Fruits and vegetables quality evaluation using computer vision: A review," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 33, pp. 243-257, 2021.
- [2] A. Goel, C. Tung, Y.-H. Lu, and G. K. Thiruvathukal, "A survey of methods for low-power deep learning and computer vision," in *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*, 2020, pp. 1-6.
- [3] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," *IEEE transactions on neural networks and learning systems*, vol. 32, pp. 604-624, 2020.
- [4] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, pp. 712-733, 2020.
- [5] S. M. McKinney, M. Sieniek, V. Godbole, J. Godwin, N. Antropova, H. Ashrafi, *et al.*, "International evaluation of an AI system for breast cancer screening," *Nature*, vol. 577, pp. 89-94, 2020.
- [6] L. Santos, F. N. Santos, P. M. Oliveira, and P. Shinde, "Deep learning applications in agriculture: A short review," in *Robot 2019: Fourth Iberian Robotics Conference: Advances in Robotics, Volume 1*, 2020, pp. 139-151.
- [7] Y. Li, "Research and application of deep learning in image recognition," in *2022 IEEE 2nd International Conference on Power, Electronics and Computer Applications (ICPECA)*, 2022, pp. 994-999.
- [8] I. Martinez-Alpiste, G. Golcarenenji, Q. Wang, and J. M. Alcaraz-Calero, "Smartphone-based real-time object recognition architecture for portable and constrained systems," *Journal of Real-Time Image Processing*, vol. 19, pp. 103-115, 2022.
- [9] X. Zhao, "Research and application of deep learning in image recognition," in *Journal of Physics: Conference Series*, 2023, p. 012047.
- [10] S. Divya, B. Adepu, and P. Kamakshi, "Image Enhancement and Classification of CIFAR-10 Using Convolutional Neural Networks," in *2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2022, pp. 1-7.
- [11] X. Zhao, P. Huang, and X. Shu, "Wavelet-Attention CNN for image classification," *Multimedia Systems*, vol. 28, pp. 915-924, 2022.
- [12] A. R. Choudhuri, B. G. Thakurata, B. Debnath, D. Ghosh, H. Maity, N. Chattopadhyay, *et al.*, "MNIST Image Classification Using Convolutional Neural Networks," in *Modeling, Simulation and Optimization: Proceedings of CoMSO 2021*, ed: Springer, 2022, pp. 255-266.
- [13] N. Saidulu, K. A. Monsley, K. S. Yadav, and R. H. Laskar, "Exploration of Deep Convolutional Neural Networks (Via Transfer Learning) for Handwritten Character Recognition," in *2022 Second International Conference on Power, Control and Computing Technologies (ICPC2T)*, 2022, pp. 1-6.
- [14] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, 2019, pp. 6105-6114.
- [15] A. Ignatov, R. Timofte, W. Chou, K. Wang, M. Wu, T. Hartley, *et al.*, "Ai benchmark: Running deep neural networks on android smartphones," in *Computer Vision – ECCV 2018 Workshops*, 2019, pp. 288-314.
- [16] Keras, "Keras Applications," <https://keras.io/api/applications/>, 2023.
- [17] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314-1324.
- [18] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
- [19] W. Wang, Y. Li, T. Zou, X. Wang, J. You, and Y. Luo, "A novel image classification approach via dense-MobileNet models," *Mobile Information Systems*, vol. 2020, 2020.
- [20] M. Kim, Y. Kwon, J. Kim, and Y. Kim, "Image Classification of Parcel Boxes under the Underground Logistics System Using CNN MobileNet," *Applied Sciences*, vol. 12, p. 3337, 2022.
- [21] L. Zhao and L. Wang, "A new lightweight network based on MobileNetV3," *KSII Transactions on Internet & Information Systems*, vol. 16, 2022.
- [22] S. Benkrama and N. E. H. Hemdani, "Deep Learning with EfficientNetB1 for detecting brain tumors in MRI images," in *2023 International Conference on Advances in Electronics, Control and Communication Systems (ICAECCS)*, 2023, pp. 1-6.

- [23] S. A. Albelwi, "Deep Architecture based on DenseNet-121 Model for Weather Image Recognition," *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, pp. 559-565, 2022.
- [24] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.
- [25] M. Shafiq and Z. Gu, "Deep residual learning for image recognition: a survey," *Applied Sciences*, vol. 12, p. 8972, 2022.
- [26] X. Wan, F. Ren, and D. Yong, "Using Inception-Resnet v2 for face-based age recognition in scenic spots," in *2019 IEEE 6th International Conference on Cloud Computing and Intelligence Systems (CCIS)*, 2019, pp. 159-163.
- [27] M. Zaman and M. S. Hamidi, "A Relative Comparison of Different CNN Models Trained on A Dataset in The Perspective of Bangladesh," in *2021 IEEE Region 10 Symposium (TENSYP)*, 2021, pp. 1-5.
- [28] A. Pujara, "Image Classification with MobileNet," *Published in Analytics Vidhya*, 2020.
- [29] Q. Xiang, X. Wang, R. Li, G. Zhang, J. Lai, and Q. Hu, "Fruit image classification based on Mobilenetv2 with transfer learning technique," in *Proceedings of the 3rd international conference on computer science and application engineering*, 2019, pp. 1-7.
- [30] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510-4520.
- [31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818-2826.
- [32] M. Wei, Q. Wu, H. Ji, J. Wang, T. Lyu, J. Liu, *et al.*, "A skin disease classification model based on densenet and convnext fusion," *Electronics*, vol. 12, p. 438, 2023.
- [33] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," *arXiv preprint arXiv:1608.08710*, 2016.
- [34] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network pruning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 11264-11272.
- [35] A. Polino, R. Pascanu, and D. Alistarh, "Model compression via distillation and quantization," *arXiv preprint arXiv:1802.05668*, 2018.
- [36] J. Yang, X. Shen, J. Xing, X. Tian, H. Li, B. Deng, *et al.*, "Quantization networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7308-7316.
- [37] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4133-4141.
- [38] S. Fu, Z. Li, Z. Liu, and X. Yang, "Interactive knowledge distillation for image classification," *Neurocomputing*, vol. 449, pp. 411-421, 2021.
- [39] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, "Hierarchical representations for efficient architecture search," *arXiv preprint arXiv:1711.00436*, 2017.
- [40] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *The Journal of Machine Learning Research*, vol. 20, pp. 1997-2017, 2019.
- [41] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251-1258.
- [42] K. Wu, Y. Guo, and C. Zhang, "Compressing deep neural networks with sparse matrix factorization," *IEEE transactions on neural networks and learning systems*, vol. 31, pp. 3828-3838, 2019.
- [43] H. Qassim, A. Verma, and D. Feinzimer, "Compressed residual-VGG16 CNN model for big data places image recognition," in *2018 IEEE 8th annual computing and communication workshop and conference (CCWC)*, 2018, pp. 169-175.
- [44] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700-4708.
- [45] J. Park, P. Aryal, S. R. Mandumula, and R. P. Asolkar, "An Optimized DNN Model for Real-Time Inferencing on an Embedded Device," *Sensors*, vol. 23, p. 3992, 2023.
- [46] Y. Gulzar, "Fruit image classification model based on MobileNetV2 with deep transfer learning technique," *Sustainability*, vol. 15, p. 1906, 2023.
- [47] Q. Sun, C. Bai, T. Chen, H. Geng, X. Zhang, Y. Bai, *et al.*, "Fast and Efficient DNN Deployment via Deep Gaussian Transfer Learning," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5360-5370, 2021.
- [48] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, "Pruning and quantization for deep neural network acceleration: A survey," *Neurocomputing*, vol. 461, pp. 370-403, 2021.
- [49] D. Passos and P. Mishra, "A tutorial on automatic hyperparameter tuning of deep spectral modelling for regression and classification tasks," *Chemometrics and Intelligent Laboratory Systems*, vol. 223, p. 104520, 2022.
- [50] K. Kamal and E.-Z. Hamid, "A comparison between the VGG16, VGG19 and ResNet50 architecture frameworks for classification of normal and CLAHE processed medical images," 2023.
- [51] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097-1105, 2012.
- [52] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84-90, 2017.
- [53] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, pp. 211-252, 2015.
- [54] J. Cui, R. Wang, S. Si, and C.-J. Hsieh, "Scaling up dataset distillation to imagenet-1k with constant memory," in *International Conference on Machine Learning*, 2023, pp. 6565-6590.
- [55] H. A. Al-Jubouri and S. M. Mahmmod, "A comparative analysis of automatic deep neural networks for image retrieval," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 19, pp. 858-871, 2021.
- [56] M. Sadhasivam, M. K. Geetha, and J. G. M. Britto, "Efficient Deep Learning Architecture for the classification of diseased plant leaves," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 33, pp. 198-206, 2024.
- [57] Y. Zhong, Z. Teng, and M. Tong, "LightMixer: A novel lightweight convolutional neural network for tomato disease detection," *Frontiers in Plant Science*, vol. 14, p. 1166296, 2023.

- [58] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 116-131.
- [59] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "Ghostnet: More features from cheap operations," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 1580-1589.
- [60] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

## Authors' Profiles



**Mohamed Abdul Careem Mohamed Raafi** is a Ph.D. candidate in Computer Science at the School of Graduate Studies, Management and Science University, Malaysia. He completed his M.Sc. in Computer Science at the University of Peradeniya, Sri Lanka, and holds a B.Sc. (Hons.) in Computer Science from the South Eastern University of Sri Lanka. He is an Associate Fellow of the Higher Education Academy (AFHEA), awarded by Auckland University of Technology, New Zealand. Currently, he is affiliated with the Department of Computer Science and Informatics at Uva Wellasa University, Sri Lanka. His research interests include deep neural networks, machine learning, artificial intelligence, intelligent systems, image classification, and Android application development. He can be reached via email at [mraafi@gmail.com](mailto:mraafi@gmail.com).



**Md. Gapar Md Johar** is Senior Vice President System, Technology and Innovation of Management and Science University, Malaysia. He is a professor in Software Engineering. He holds Ph.D. in Computer Science, M.Sc. in Data Engineering and B.Sc. (Hons) in Computer Science and Certified E-Commerce Consultant. He has more than 40 years of working and teaching experience in various organizations include Ministry of Finance, Ministry of Public Enterprise, Public Service Department, Glaxo Malaysia Sdn Bhd and Cosmopoint Institute of Technology. His research interests include learning content management system, knowledge management system, blended assessment system, data mining, RFID, e-commerce, image processing, character recognition, data analytics, artificial intelligent, and healthcare management system. He can be contacted at email: [mdgapar@msu.edu.my](mailto:mdgapar@msu.edu.my).

**How to cite this paper:** Raafi Careem, Md Gapar Md Johar, "An Optimized Deep Neural Network Model for Image Classification in Resource-constrained Environments", *International Journal of Intelligent Systems and Applications(IJISA)*, Vol.18, No.2, pp.64-84, 2026. DOI:10.5815/ijisa.2026.02.05