UNIVERSITY OF BIRMINGHAM

MSc. Project

Channel-overlapping group convolution: a simple yet effective design for lightweight CNNs

Submitted in conformity with the requirements
for the degree of MSc. Computer Science
School of Computer Science
University of Birmingham

Haotian Li
Student ID: 2424400
Supervisor: Dr Jianbo Jiao

September 2023

Acknowledgements

Over three months have been spent to accomplish and carve this project, during the entire procedure of this project, my knowledge bank has been extremely expanded, and I learned tons of things during this wonderful period.

I would like to offer my major gratitude to Dr Jianbo Jiao, who is the significant person to assist and supervise me in finishing this project. It is not possible that I could implement my idea in this project without him. He offered me a lot of advice which helped me to enrich my project. I think Dr. Jiao not only taught me the research methods which can be used to solve the research problems in my project and explore the reasons behind the experiment results, but also taught me how to discuss and solve academic problems from a scientific perspective. I sincerely thank Dr. Jiao for all the help he has provided.

I would also like to thank the University of Birmingham for offering me sufficient teaching and academic resources. Thanks to the all efforts of the staff and teachers at the University of Birmingham, I was able to spend such a meaningful year.

The idea of this project has been in my mind for quite a long time, I am very glad that my idea can finally become this project, it is meaningful to me. Thanks to everyone who helped me to complete this project.

Abstract

In this project, we present an efficient structure called channel-overlapping group convolution, which mainly focus on improving the efficiency of lightweight convolutional neural network. The channel-overlapping group convolution is modified based on the group convolution. It alleviates the negative impact of group convolution while retaining the feature that group convolution can reduce computational complexity. We introduce this structure into the expansion procedure in the inverted residual block to optimize the expensive pointwise convolution. We replaced the original inverted residual block in MobileNetv2 with the improved block to obtain the improved network, and then, explored the performance improvement from our structure in three image tasks: image classification, object detection, and semantic segmentation. Based on the experiment results, we found that introducing channel-overlapping group convolution can effectively decrease the model size and gain solid performance improvement on most datasets used in this project. We then further studied the dataset with poor network performance and summarized two preconditions for using this structure: 1) The dataset should not follow the long-tail distribution with the excessively long tail area. 2) The model parameters should be set reasonably.

Key words: Lightweight convolutional neural network, group convolution,

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Convolutional neural networks (CNNs) have achieved great success in the realm of computer vision. Classical CNN models such as VGG [23] and Resnet [8] had made significant contributions to the development of CNN, and in recent work, ConvNext [16] had achieved outstanding performance as a pure convolutional neural network. However, although these networks have excellent performance, they generally require expensive computational costs. In this concern, a type of network focusing more on being lightweight and efficient has been designed, known as lightweight neural networks. The lightweight models are mainly designed for resource-constrained devices, such as mobile phones and embedded systems. Recently, there have been many lightweight convolutional neural networks which had shown some outstanding performance in the computer vision realm, such as the MobileViT series [19, 25, 20], EfficientViT [15], etc. These lightweight models enjoy high efficiency and small model sizes. In recent research on lightweight networks, the majority of studies had been related to hybrid networks, which combine CNNs and vision transformer (ViT) [4].

The hybrid networks mainly focus on how to efficiently integrating the strengths of CNN and ViT. Benefiting from the spatial inductive biases in CNN structures and the self-attention mechanism in ViT structures, hybrid models generally have the strong ability to capture both local and global features in the image. However, due to the high computational cost of ViT and the fact that visual models incorporating ViT often require strong

data augmentation and L2 regularization [19], hybrid models theoretically have a high model size and training cost, which is not in line with the concept of lightweight networks. Therefore, the optimization of the hybrid model mainly focuses on the self-attention structure in ViT. For example, in MobileViT [19], standard convolutions were added before and after self-attention, and the 'fold' operation is designed to reduce the computational complexity of the ViT module. Additionally, in Efficientvit [15], the cascaded group attention was developed to reduce the computational cost of attention structure.

However, most hybrid networks optimize the self-attention structure and do not make too many changes to the CNN structure. For example, in the structure of MobileViT [19], the inverse residual structure of MobileNetv2 [22] is basically directly introduced to the early stage of network architecture to filter the local information without making too many changes. However, according to ShuffleNet [27], the pointwise convolution layers in the inverted residual block contain expensive computational costs, which can be further optimized. Shufflenet introduced pointwise group convolution to optimize the computational complexity of pointwise convolution, but simply introducing group convolution can reduce the performance of the network [27]. In order to better eliminate the negative impact of group convolution, we have designed the channel-overlapping group convolution to replace the traditional group convolution. In this project, our main contributions are designing the channel-overlapping group convolution and introducing channel-overlapping group convolution into the inverted residual block of Mobilenetv2 [22]. Based on our inference and experiment results, it is proved that this structure can effectively reduce the computational complexity, model size, and memory access cost. Simultaneously, the performance of the network can be improved by introducing our structure, as long as the preconditions can be met. We also discussed the advantages and disadvantages of this structure to determine the preconditions of using this structure and the problems that needed to be further researched.

Chapter 2

Literature review

2.1 Network structure evolution

Convolutional neural networks have achieved huge success in the computer vision realm. By continuing to refine the network architecture, there are many breakthroughs in convolutional neural networks providing considerable contributions and effective design ideas. VGG nets [23] designed the architectures focusing on depth, which achieved excellent accuracy performance. Instead of applying a large 7×7 convolution kernel size, VGG nets introduced a series of small 3×3 convolutional filters to not only decrease the volume of parameters under the similar receptive fields (three layers 3×3 convolution filters have the effective receptive fields of a single 7×7 convolution filters layer) but also impose regularization by decomposing the large convolution filters and injecting non-linearity within. The entire network architectures of VGG nets are simple but effective, which provides a significant idea about using deeper layers of small convolution filter size to replace the large and inefficient convolutional layer. ResNet [8] is a significant breakthrough work after VGG nets. One of the major contributions of Resnet is breaking the limitations of previous convolutional neural networks in the depth dimension. Before ResNet, blindly increasing the depth of the convolutional network will raise the problems of vanishing/exploding gradients and degradation [8], which will interfere with model convergence and decrease the model performance. Therefore, the deep convolutional networks gener-

ally contained at most dozens of layers at the early stage. However, ResNet developed the residual concept to improve the network architecture and dramatically increased the depth of the network to a maximum of 152 layers by combining with batch normalization layers. Specifically, Resnet added a shortcut connection, which is adding the input feature to the output feature as the final output, on every several layers to allow the network to focus on optimizing residual mapping, which is easier than optimizing original mapping. From the experiment results, applying residual learning will significantly address the degradation problem, moreover, by applying more layers with bottleneck blocks, ResNet achieved obvious accuracy improvement with over 100 layers. The residual structure was so effective that became one of the most famous structure designs of convolution neural networks.

2.2 Light-weight neural network development

The lightweight neural networks are mainly designed to be loaded on resource-constrained devices; therefore, they focus more on improving network efficiency instead of introducing large and complex structures to achieve outstanding performance. The problem of researching lightweight neural networks can be determined as: How to minimize the computational cost with the lowest performance loss? The improvement of many lightweight neural networks, such as MobileNet series [10, 22, 9], ShuffleNet series [27, 18], and MobileViT series [19, 20, 25], mainly focuses on how to descend the volume of parameters and execution latency, which renders the lightweight neural networks contain the properties of low model size, low computational cost, and low latency.

2.2.1 MobileNets series

MobileNet [10] is one of the convolutional networks which developed the lightweight structure at the early stage. To decrease the computational cost, MobileNet introduced depthwise separable convolution to factorize the standard convolution. Depthwise separable convolution contains the depthwise convolution, which is a set of 3×3 convolutional fil-

ters applied on each channel of input feature, and pointwise convolution, which is the 1×1 convolutional filter. The comparison between standard convolution and depthwise separable convolution is shown in Figure 2.1. This structure allows the network to dramatically reduce the computational cost (around 8 to 9 times smaller than that of standard convolutions) with an acceptable performance drop.

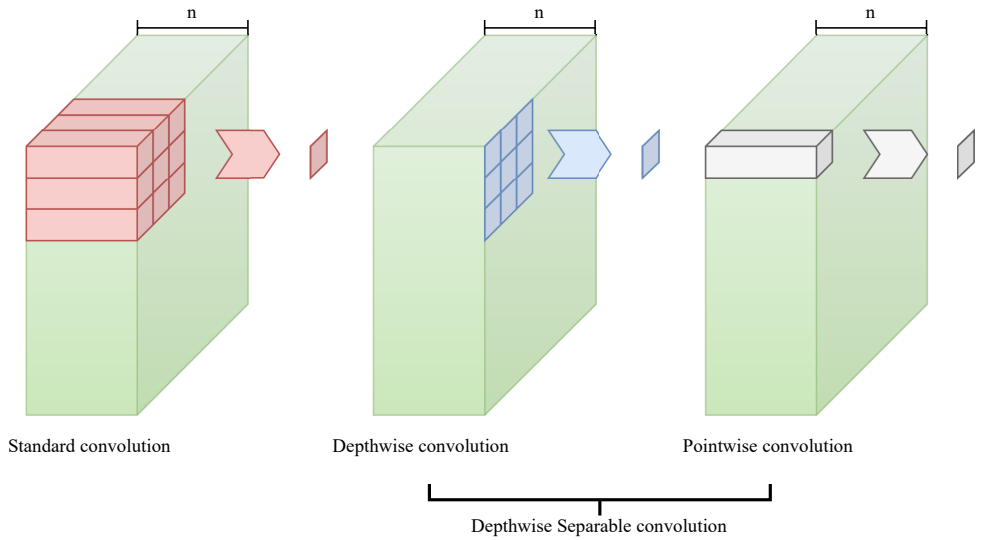


Figure 2.1: Standard convolution and Depthwise Separable convolution.

The lightweight structure of MobileNet was further improved by MobileNetv2 [22]. Inspired by Resnet [8], MobileNetv2 invented a significant lightweight architecture called inverted residual structure, which remained the idea of shortcuts from the residual block and introduced expansion factor and depthwise separable convolution. Specifically, the inverted residual block has two main differences compared with the residual block. Firstly, unlike the residual block, they deployed pointwise and depthwise convolution layers to construct the entire block, which effectively reduced the computational cost compared with the original residual block. Secondly, rather than mapping features to lower dimensions, mapping the input features to higher channel dimensions showed better efficiency and performance when applying depthwise separable convolutions, therefore, the expansion factor was introduced to determine the channel dimension expansion rate of input features. In MobileNetv2, the expansion factor is 6, which means the number of channels for input feature maps will be increased by 6 times. The inverted residual block applied

pointwise convolution to scale up the channel dimension of input features and then used depthwise convolution to filter the features. In the end, deploying pointwise convolution again to project back to the output channel number and add with the input feature to gain the final output. The inverted residual block is shown in Figure 2.2. Apart from that, the non-linear activation was ReLU6 instead of ReLU to improve the robustness of the network, and the inverted residual block directly applied linear project in the last pointwise convolution since the ReLU will inevitably lose information in low dimension. When pointwise convolution filters the feature map from high to low dimension, applying ReLU transformations will lead to information loss. For object detection tasks, MobileNetv2 also modified the Single Shot Detector (SSD) [14] architecture. All the standard convolutions in the SSD were replaced by separable convolutions which are depthwise convolutions followed by pointwise projection and the backbone was replaced with MobileNetv2. The improved SSD is called SSD-Lite which enjoyed less volume of parameters and better efficiency than SSD. Inverted residual block is an effective lightweight structure which still is widely used in recent lightweight network development, such as MobileViT.

MobileNetv3 [9] made further improvements in both performance and efficiency aspects by introducing the attention mechanism and h-swish non-linear activation. The Squeeze-and-Excite attention has been introduced into the inverted residual structure in MobileNetv2 and is added after the depthwise convolution. From the results, this structure improves the efficiency of the structure, which allows the model to eliminate redundant layers. Specifically, the Squeeze-and-Excite attention is applying average pooling for each channel and using two full connections to process the pooling feature, in the end, multiplying the original feature map and pooling feature to get the new feature map. The structure is shown in Figure 2.3. Apart from that, a new nonlinearity called h-swish has been introduced, which enjoys the advantages of optimized implementations, eliminating numerical precision loss, and decreasing the latency cost. The h-swish activation function can be described as the formula below:

$$hswish[x] = x\frac{ReLU(x+3)}{6}$$

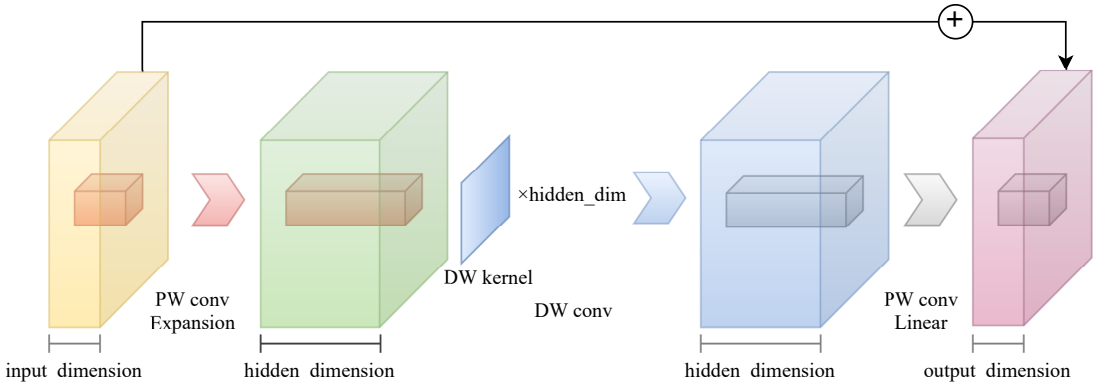


Figure 2.2: Inverted residual block of MobileNetv2

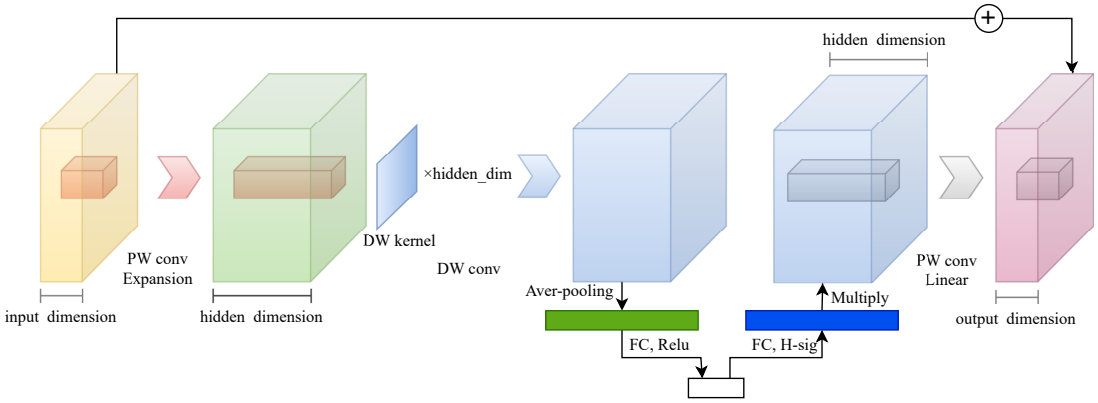


Figure 2.3: MobileNetv3 block

2.2.2 ShuffleNet series

ShuffleNet [27] is a lightweight convolution neural network improved based on MobileNet [10]. The improvement was mainly focused on the channel dimension. ShuffleNet found the efficiency limitation from pointwise convolution in depthwise separable convolution. The pointwise convolution required considerable complexity; therefore, they introduced pointwise group convolution by adding the group concept to replace the pointwise convolution and descend the computational cost. The difference between pointwise convolution

and pointwise group convolution is shown in the Formula below:

$$\frac{Cost_{pw}}{Cost_{pgw}} = \frac{D_F^2 \times N \times M}{D_F^2 \times \frac{N}{g} \times \frac{M}{g} \times g}$$

Which $Cost_{pw}$ and $Cost_{pgw}$ represent the computational cost of pointwise convolution and pointwise group convolution, D_F^2 represents the size of feature map, N and M represent the output and input channel number. It is obvious introducing the group convolution concept will decrease the computational cost. However, directly applying group convolution will cause another side effect which is the output feature will only map the features from a single group of the input feature map, and will lose the information from other groups. To address this issue, ShuffleNet used channel shuffle to reconstruct feature maps obtained from pointwise convolution to reduce the side effects caused by introducing group convolution. ShuffleNet achieved better efficiency by constructing the network based on pointwise group convolution and channel shuffle.

ShuffleNetv2 [18] is the improved version of ShuffleNet, they argued that designing the lightweight neural network cannot only focus on the indirect metric of computational complexity, such as floating point operations (FLOPs), the direct metric, speed for example, should be considered. One of the factors which mainly impact the speed is memory access cost (MAC) which is the major target of ShuffleNetv2 trying to optimize. Four significant guidelines for optimizing network architecture and improving the network speed were discussed in ShuffleNetv2.

G1) Equal channel width minimizes memory access costs. This guideline can be proved by formulas. Because we mainly focus on the pointwise convolution, which the kernel size is 1, so in the formula, kernel size is not been considered. Let h and w represent the size of the feature map, c_i represents the number of input channels and c_o represents output channels. The memory access cost (MAC) can be shown as:

$$MAC = hwc_i + hwc_o + c_ic_o,$$

which MAC is equal to the sum of the input feature map (hwc_i), output feature map (hwc_o), and the kernels c_i c_o. The volume of FLOPs of pointwise convolution can be presented as:

$$FLOPs = hwc_ic_o.$$

Therefore, the relationship between MAC and FLOPs can be represented as:

$$MAC \geq 2\sqrt{hwFLOPs} + \frac{FLOPs}{hw}.$$

So, the lowest bound of MAC is related to the FLOPs and only when the input channel is equal to the output channel, the MAC will reach the lowest bound. In a word, within the same FLOPs, when $c_i : c_o$ is approaching 1:1, the MAC becomes smaller and the speed of the network evaluation is faster.

G2) Excessive group convolution increases MAC. Under the condition of same FLOPs ($B = \frac{(hwc_ic_o)}{g}$) and fixed input feature map hwc_i, the MAC can be represented as:

$$MAC = hwc_i + \frac{Bg}{c_i} + \frac{B}{hw}.$$

Clearly, under the same FLOPs, MAC will increase if the number of groups increased. Therefore, it is unwise to use a large number of groups simply when designing the network architecture.

G3) Network fragmentation reduces the degree of parallelism. Network fragmentation can improve the accuracy, but in ShuffleNetv2, it is believed that such a fragmented structure will decrease the efficiency of GPU with strong parallel computing capability, and incurs extra overheads like kernel launching and synchronization. The specific efficiency-dropping magnitude of increasing network fragmentation will depend on certain factors, such as, serial fragmentation, parallel fragmentation, type of platform, and number of channels.

G4) Element-wise operations are non-negligible. The element-wise operators, like

ReLU, AddTensor, and AddBais, occupy a considerable amount of time since they generally lead to high memory access costs.

Overall, these 4 guidelines are provided for designing a more efficient neural network, and ShuffleNetv2 was developed based on them and achieved better performance than the last generation's lightweight convolutional neural network at that time.

2.2.3 Hybrid lightweight neural network

Recent research on lightweight networks has mainly focused on the combination of convolution and Vision transformers (ViT). This is because ViT contains the self-attention mechanism to effectively process global information, but pure ViT networks require a considerable volume of parameters to achieve good performance since pure ViT-based models lack the image-specific inductive bias from the convolutional neural network [4]. To improve the efficiency of the network, hybrid networks combining ViT and convolutional structures are developed. For lightweight neural networks, MobileViT [19] and EfficientViT [15] are the recent hybrid lightweight neural networks which achieve excellent performance. In MobileViT, MobileNetv2 blocks were deployed at the front end of the network structure, and MobileViT blocks were deployed at the middle and bottle stages of network architecture. MobileViT block contained three parts: local representations, global representations, and fusion. Local representations consist of a standard convolution and a pointwise convolution to filter the feature. Global representations were the modified transformer block. Fusion deployed pointwise convolution to project the feature map and standard convolution to filter the feature map which concatenated projected and input feature map. The MobileViT architecture enjoys the benefits both from ViT and CNNs, by using the MobileNetv2 block, it remains the lightweight properties with reasonable performance.

2.3 Channel-wise modification in convolutional neural network

Making changes to the channel dimension of the model has always been a good option for the development of convolutional neural networks. The ShuffleNet [27], which is discussed in section 2.2.2, introduced pointwise group convolution and channel shuffle to improve the network efficiency. Additionally, the Squeeze-and-Excitation (SE) Network [11] is another popular convolutional neural network which focuses on channel-wise modification. SE block is a computational unit which can be directly embedded in other network architecture. The major goal of the SE block is to help the network effectively fetch global information. SE block contains two important steps, 'squeeze' and 'excitation'. The 'squeeze' operation uses global average pooling to exploit the global information from each channel, the 'excitation' operation uses two fully connected layers followed by nonlinearities of ReLU and Sigmoid to gain the scalar. This scalar will be multiplied by the original feature map to obtain the feature map processed by the SE block. The benefits of SE block are, first, it helps pure convolutional networks better process global information, second, SE block is flexible enough to directly embed into the majority of convolutional networks, third, SE block is an efficient choice to improve the network performance with a slight volume of parameters increasing.

Chapter 3

Background

In this section, the background knowledge related to this project is introduced.

3.1 Convolution Neural Network (CNN)

CNN has already achieved huge success in the computer vision realm. Convolutional neural networks have great advantages in image processing. In many visual-related tasks, such as image classification, image segmentation, image generation, etc., convolutional neural networks have mostly achieved good results.

The basic convolutional layer can be described as the structure in Figure 3.1 (ignoring channel dimension). The convolution kernel contains the weights of the layer and is used to filter the feature of the input map to get the output feature map. The convolution operation can be seen as shifting the convolution kernel, and making dot products with related areas of feature map, this process is similar to the stimulation of receptive field regions in biological vision.

Benefiting from this special design, convolution neural networks are based on the weight-shared architecture and are good at handling image information. Various kinds of CNNs are developed to well solve many different image tasks and provide solid contributions in the computer vision realm.

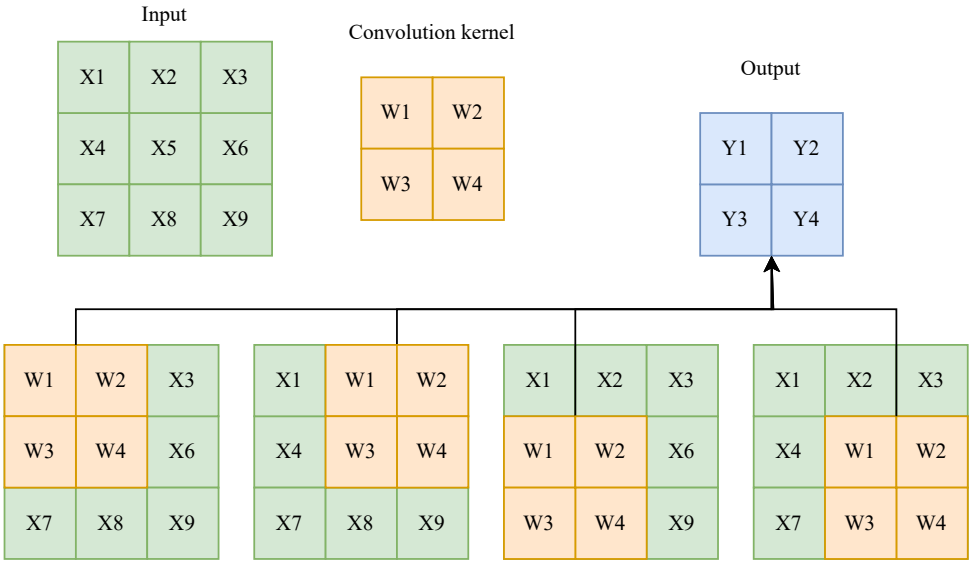


Figure 3.1: Convolution operation

3.2 Non-linearity activation

The aim of deploying active functions is to introduce non-linearity to neural networks, which allows neural networks to better solve complex problems. The traditional activation function is sigmoid, which frames the value between 1 to -1. As the evolution of activation functions, more simplified or better-performing non-linearities, e.g., ReLU (Figure 3.2), were developed to decrease the computational cost and improve the performance of networks. In this project, we followed the improvement from MobileViT [19], introducing SiLU activation functions to improve the MobileNetv2 block.

3.3 Optimizer

Optimizers of neural networks are used to optimize the target function (loss function) solving process of neural networks. Optimizers are designed not only for quickly converging the loss value of networks but also for avoiding the local optima and helping the network to accurately find the global optimal solution. One of the fundamental optimizers is Stochastic Gradient Descent (SGD), using the momentum method could further optimize the SGD optimizer. In this project, AdamW [17], which is the improved version

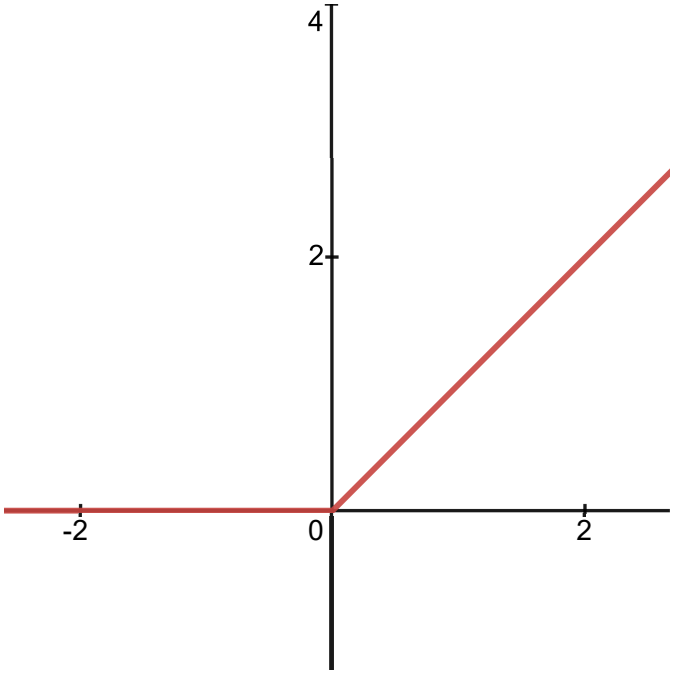


Figure 3.2: Non-linearity of ReLU.

of Adam, is deployed to optimize the training procedure.

3.4 Learning rate scheduler

The learning rate scheduler (LR scheduler) provides the dynamic learning rate for network training procedures. This strategy is useful when the network is solving comparatively complex problems. Generally, the LR schedulers are deployed to help the network escape from local optima and optimize the convergence process during the different stages of training. Specifically, the learning rate can be large at the early stage of the training procedure to accelerate the convergence speed, however, the large learning rate could cause the network to ignore the global optimal solution at the final stage of training, therefore, the learning rate should be generally decreased to the suitable value as the increase of iterations. Some LR schedulers contain the warm-up strategy, which is after the learning rate reaches the set value, it will be increased to jump out the local optima. This strategy can effectively prevent the situation that the model converges to the local optimal solution.

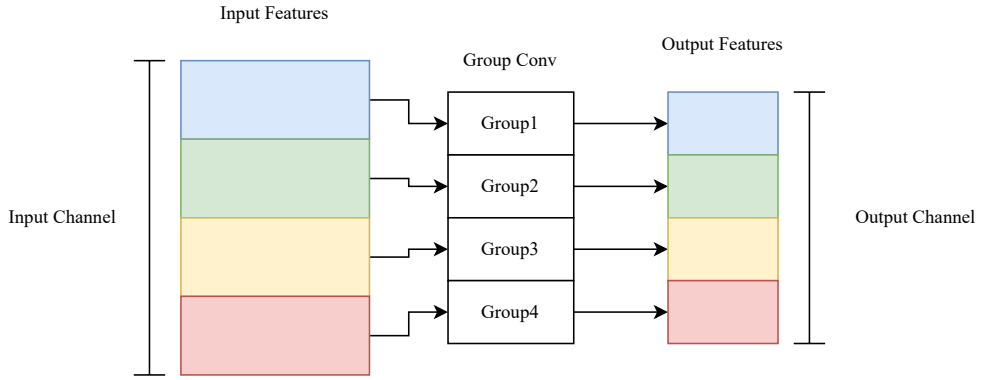


Figure 3.3: Group convolution (4 groups).

3.5 Group convolution

Group convolution is a variation of convolution operation. Group convolution can effectively decrease the computational cost and volume of parameters. According to EfficientNet [24], introducing group convolution can improve the network capability of process fine-grained features. The structure of group convolution (which the group number is 4) is shown in Figure 3.3. The group convolution first divides the input feature map into several groups, then filters each group by individual group convolution kernel to get the output group, and finally, concatenates all output groups to get the output feature map. By deploying group convolution, the floating-point operations (FLOPs) can be decreased. However, directly replacing standard convolution with group convolution will cause the performance to drop. This is because each group of the output feature map only contains the information from a single group of the input feature map, in a word, group convolution will create information isolation related to the channel dimension [27].

3.6 Data augmentation and regularization

Data augmentation and regularization are the methods that are applied in network training procedures to improve the performance of the network. Specifically, data augmentation and regularization are used to prevent overfitting of networks. Data augmentation focuses on enhancing the training dataset. Data augmentation includes various methods,

such as randomly cropping and rotating images, as well as randomly covering certain areas of the image, to enhance the robustness of the model, and reduce overfitting. There are many advanced data augmentations, such as RandAugment [2], mixup [26], and Random erasing [28], which can provide solid improvement for networks.

Methods that can reduce overfitting are called regularization. Among them, L2 regularization is widely used in machine learning. The principle of L2 regularization is adding a special penalty after the loss function to prevent the model weight from overfitting the training data.

3.7 Image tasks

3.7.1 Image Classification

Classification tasks can be seen as the fundamental task of the computer vision realm and a significant task to evaluate the performance of networks. To solve the classification problem, the training dataset should contain the training images and the related labels. For binary classification problems, the labels can be the simple 1 and 0. The multi-class classification problem can be transformed into a binary classification problem for each category, which is to distinguish whether the image belongs to this category or not.

For binary classification, the confusion matrix contains 4 types of results, which are TP (true positives), FP (false positive), FN (false negatives), and TN (true negatives). True and false represent whether the predictions are correct or not, and positive and negative represent the prediction results of the classifier. The accuracy of binary classification can be defined as:

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

However, when the positive and negative samples are severely uneven, then accuracy cannot effectively evaluate the performance of networks. Based on the confusion matrix, the other two metrics which are precision and recall are widely considered. They can be

described as:

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

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

F-score is another evaluation method calculated from precision and recall. A general F-score is $F_\beta - score$, which can be described as:

$$F_\beta - score = (1 + \beta^2)\frac{precision \cdot recall}{\beta^2 \cdot precision + recall}$$

This evaluation method can better evaluate the network when the training samples are distributed unevenly.

For the multi-class classification task, the confusion matrix, which is shown in Figure 3.4, should record the relationship between the labels and prediction results. The accuracy is the ratio of correct predictions to the total number of validation samples.

		Prediction			
		A	B	C	Total
Label	A	15	3	2	20
	B	9	15	6	30
	C	1	1	48	50
	Total	25	19	56	100

Figure 3.4: An example of the confusion matrix of muti-class classification task, there are three classes (class A, class B, and class C) and 100 samples in the example. Prediction means the prediction from network, and label means the true label of dataset.

3.7.2 Object Detection

The purpose of object detection is to correctly frame the objects in the image using the bounding boxes. Generally, the final output of the network should at least contain the 5 values, which include 4 values to determine the position and size of the bounding box, and 1 value to determine the category. There are generally two methods to determine the bounding box: 1) Using two values to represent the central point of the bounding box on the x-axis and y-axis, and using the remaining two values to represent the length and width of the box. 2) 4 values represent the coordinates of the top right and bottom left vertices of the bounding box. For this project, we use the second method as the data form of a bounding box.

The evaluating indicator of the object detection task is mean Average Precision (mAP). Mean average precision is the mean value of the average precision. To calculate the average precision in the object detection task, intersection over union (IoU) needs to be considered. IoU can be defined as:

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$

The TP and FP are defined based on the threshold of IoU, in this project, the threshold is 0.45. The bounding box which's IoU higher than the threshold is considered as TP, otherwise, it will be considered as FP. Then, the precision and recall can be calculated based on the formula mentioned before. The average precision of each class is the area of the P-R curve composed of precision and recall. And calculating the mean value of the average precision of all categories.

3.7.3 Semantic Segmentation

The segmentation task is to group the pixels in a localized image by creating a segmentation mask. Semantic Segmentation can be seen as the classification task for every pixel in the image, therefore, it is similar to the multiple classification task. In this project, the labels of the semantic segmentation task are formed as single-channel PNG pictures, and

(a) Segmentation label (b) Image sample

Figure 3.5: An example of segmentation label. The left picture is the segmentation label of the right picture. The label and image are from Pascal VOC 2012 [6], file name: 2011_002644

the colour of each pixel in this kind of picture is determined by a value between 0 to 255 (this value also represents the category of pixel). The Figure 3.5 shows an example of a semantic segmentation label.

Usually, mean intersection over union (mIoU) is used to evaluate the performance of the model in semantic segmentation tasks. The mIoU is the mean value of IoU for all categories. The areas within IoU are calculated depending on the number of pixels, which can be described as:

$$IoU = \frac{Pixels\ Overlap}{Pixels\ Union}$$

$$mIoU = \frac{1}{c}\sum_{n=1}^{c} IoU_n$$

Where the $Pixels_overlap$ and $Pixels_union$ represent the number of pixels of overlap area and union area, c represents the number of classes, IoU_n represent the IoU of nth class.

Chapter 4

Methodology

This chapter first discussed the detail structures of channel-overlapping group convolution (section 4.1) and modified inverted residual block (section 4.3). Afterwards, the architecture of improved network is introduced in section 4.3. Finally, the implementations of networks for Object Detection task (section 4.4) and Semantic Segmentation task (section 4.5) are discussed.

4.1 Channel-overlapping group convolution

Channel-overlapping group convolution is improved based on group convolution. It is mainly applied to replace the first pointwise convolution in the inverted residual block of MobileNetv2 [22]. Pointwise convolution is considered to have high computational complexity [27], therefore optimizing the pointwise convolution in the inverted residual block may effectively improve model efficiency. However, as stated in ShuffleNet [27], directly applying group convolution to pointwise convolution can result in the model losing some of the information on the channel dimension, thereby reducing model accuracy. To address this issue, we have incorporated the concept of channel-overlapping group convolution into the inverted residual block. Channel-overlapping group convolution can enhance the information connectivity of feature maps in the channel dimension, alleviate the feature isolation between different groups caused by traditional group convolution,

and reduce computational complexity. To our best knowledge, the most similar to our work is the pointwise group convention and channel shuffle proposed by ShuffleNet, which used group pointwise convolution to reduce the computational complexity, however, the ShuffleNet unit cannot be simply embedded in MobileNetv2 block, and the channel shuffle will spend the non-negligible amount of inference time [18].

Compared to traditional group convolution, channel-overlapping group convolution first groups the input feature maps on the channel dimension and then rearranges them, so that each group convolution can filter the feature maps obtained by concatenating two sets of feature maps. In this structure, each group of output feature maps can contain feature information from neighbouring groups, thus alleviating the adverse impact of group convolution on information isolation in the channel dimension.

Channel-overlapping group convolution contains two fixed parameters: the number of groups and overlap groups. Considering the structural specificity of this group convolution, the number of groups should be above 2, as channel overlapping convolutions with a group number of 2 are equivalent to standard convolutions. Since improving the MobileNetv2 block is the major goal, setting the number of groups to 3 is not a suitable choice, since in the structure of MobileNetv2, the number of channels in many layers cannot be evenly divided by 3. In this regard, 4 and 8 seem to be the suitable number of groups for channel-overlapping group convolution. However, according to the guidelines from ShuffleNetv2, using a large number of groups and increasing the fragmentation of the network will reduce the speed and efficiency of the network, therefore, dividing the input feature into 4 groups is the reasonable choice for channel-overlapping group convolution. When the number of groups is determined to be 4, the corresponding number of overlapping groups is limited to 1 and 2, because the number of overlapping groups cannot exceed 2, and being less than one group is equivalent to traditional group convolution. From the experiment results in the table 4.1, we found that the networks perform better when only have a single overlap group. In conclusion, to better improve the MobileNetv2 block, channel-overlapping group convolution will split the input feature into four groups,

and rebuild them to make sure each group contain a single overlap group with a neighbour group, the structure is shown in Figure 4.1, and improved pointwise convolution F_{cog} can be described as:

$$x = Concat(x_1, x_2, x_3, x_4)$$

$$F_{cog} = F_{gp}(Concat(x_1, x_2, x_2, x_3, x_3, x_4, x_4, x_1))$$

Where x represents the input feature map, x_n represents the related group of the input feature map, F_{gp} represents pointwise group convolution which the number of groups is 4. *Concat* represents concatenate operation.

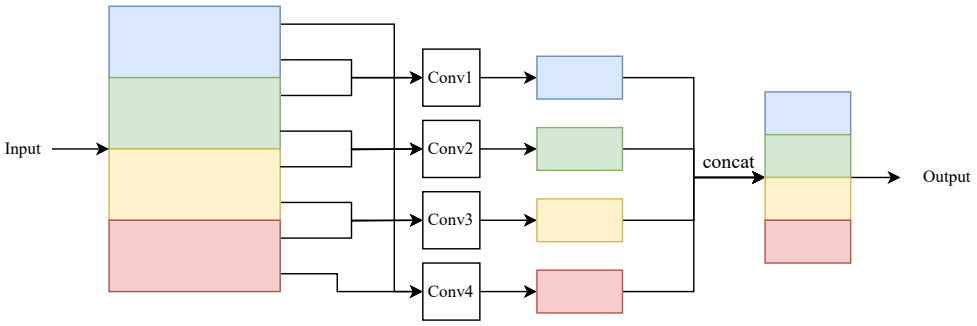


Figure 4.1: Channel-overlapping group convolution

	e2	e4	e6
one overlap group	**86.3%**	**86.8%**	**85.7%**
two overlap groups	83.5%	84.1%	83%

Table 4.1: The Top-1 accuracy of models with one overlap group and two overlap groups on the flowers dataset. e2, e4, e6 represent expansion factor equals 2, 4, and 6.

4.2 Improved MobileNetv2 Block

The modification mainly focuses on the pointwise convolutions of the MobileNetv2 block (MV2 Block). According to the guidelines of ShuffleNetv2 [18], we have decided to improve the first pointwise convolution of the mv2 block with the concept of channel overlapping group convolution. The benefits of this improvement are, first, decreasing the FLOPs required and reducing the computational complexity, second, according to the second

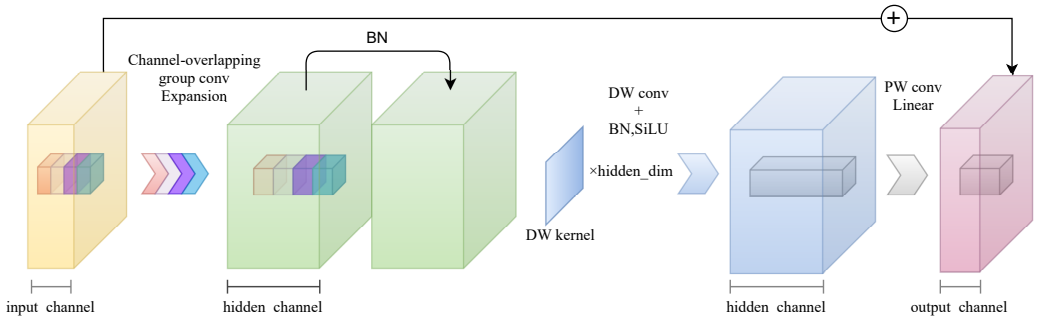


Figure 4.2: Improved inverted residual block.

guideline in ShuffleNetv2, under similar FLOPs, the closer the ratio of input channel to output channel is to 1, the smaller the MAC, and the faster the model speed, therefore, introducing this new structure will help the model have smaller MAC and lower latency under the same conditions. Specifically, in the original MobileNetv2 Block, the first pointwise convolution maps the input feature to the higher dimension, and the expansion factor is 6, which means the ratio of the number of output channels to the number of input channels is 6, however, by introducing our work, for each group of convolutions, the ratio will decrease to 3, which is more approaching to 1. Moreover, if we decrease the expansion factor to 2, then the ratio of the output channel to the input channel is 1, which achieves the lowest MAC and theoretically lowest latency. We decided to not introduce this new concept into the second pointwise convolution (linear project) in the MobileNetv2 block, this is because contrary to the previous introduction of channel overlapping group convolution to reduce MAC, introducing this structure in the linear projection of MobileNetv2 block will increase the ratio of input channel number to output channel number, increase MAC, and thus increase latency.

The improved MobileNetv2 block was implemented as: first, using pointwise convolution, which introduces channel-overlapping group convolution, to expand the channel dimension of the input feature map, then, after batch normalization, depthwise convolution is applied to filter high-dimensional features, finally, after another batch normalization layer and SiLU nonlinearity, pointwise convolution is used to project to the output feature map. The structure of the improved block is shown in Figure 4.2.

4.3 Network structure

To prove the effectiveness of channel-overlapping group convolution, the improved network basically inherits the original structure of MobileNetv2. Apart from the improved inverted residual block, another modification is the inverted residual setting of the first block. In MobileNetv2, the first inverted residual block did not expand the channel dimension, therefore, the first pointwise convolution layer was deleted. In fact, based on our experiment results, which are shown in the table 4.2, adding the first pointwise convolution (expansion procedure) in the first original block cannot offer solid performance improvement for the network, under certain situations, it may incur the performance dropping. However, for the improved block, we found adding the pointwise convolution in the first block would further improve the performance, therefore, we decided to use the version of network architecture which adds the pointwise convolution in the first block.

	MobileNetv2					
	e2		e4		e6	
Add expansion	✘	✔	✘	✔	✘	✔
Flowers dataset	84.6%	**84.9%**	**86%**	83.5%	85.2%	**85.4%**
Caltech-101	**82.8%**	82.3%	**84.6%**	83.3%	84.5%	84.5%
	Improved Network					
	e2		e4		e6	
Flowers dataset	84.3%	**86.3%**	86.8%	86.8%	85.4%	**85.7%**
Caltech-101	83.2%	**83.8%**	84.8%	**85.4%**	84.6%	**84.9%**

Table 4.2: The results of MobileNetv2 and improved network before and after adding the expansion procedure in the first block of network.

The expansion factor is considered as the major parameter to determine the size of the network, expansion factor can be set to 2, 4, and 6 to build the small, medium, and standard network size. Relatively, MobileNetv2 also sets three network sizes with expansion factors of 2, 4, and 6 for comparison. The implementing details of the improved network structure are shown in the table 4.3.

Structure name	e	c	n	s	Input size
Standard conv	-	32	1	2	$224^2 \times 3$
Improved Block	2/4/6	16	1	1	$112^2 \times 32$
Improved Block	2/4/6	24	2	2	$112^2 \times 16$
Improved Block	2/4/6	32	3	2	$56^2 \times 24$
Improved Block	2/4/6	64	4	2	$28^2 \times 32$
Improved Block	2/4/6	96	3	1	$14^2 \times 64$
Improved Block	2/4/6	160	1	2	$14^2 \times 96$
Improved Block	2/4/6	320	1	1	$7^2 \times 160$
conv2d 1×1	-	1280	1	1	$7^2 \times 320$
avgpool 7×7	-	-	1	-	$7^2 \times 1280$
conv2d 1×1	-	classes	-		$1 \times 1 \times 1280$

Table 4.3: The network architecture of improved network. e represents expansion factor, c represents output channel, n represents that the stage will be repeated n times, s represents stride.

4.4 Object Detection

Both MobileNetv2 and the improved network need a frame to accomplish image tasks other than classification tasks. They will serve as the backbone networks and cooperate with the object detection architecture to complete the image object detection task. The SSD-Lite [22] is the object detection architecture frame to be used to compare the object detection performance between MobileNetv2 and the improved network. SSD-Lite is the lightweight version of SSD [14], which was came up by MobileNetv2. SSD-Lite replaced all the standard convolutions in SSD with separable convolutions, which contain depthwise convolution and pointwise convolution, to reduce the parameters volume, computational cost, and inference time. Considering the improved model only replaces the block structure and retains the original layers design, therefore, the way of embedding the backbone network into SSD-Lite is similar to MobileNetv2.

4.5 Semantic segmentation

DeepLabv3 [1] is deployed in this project as the network frame to achieve the semantic segmentation task. According to MobileNetv2 [22], when the ratio of input image spatial

resolution to final output resolution is 16, the model enjoys good efficiency, therefore, the comparison of MobileNetv2 and the improved network will follow this ratio. The backbones will be a part of the network which is the first 5 stages (which the output feature map size is 6 times smaller than the input size) of MobileNetv2 and improved networks. The backbone is followed by the DeepLabv3 head. We choose to implement the Atrous Spatial Pyramid Pooling module (ASPP), which contains three standard dilate convolutions with different dilate rates, one 1×1 convolution, and one image pooling branch, in the DeepLabv3 head.

Chapter 5

Experiment

This section will mainly introduce the details of the experiment, including the data, parameters configurations, etc. Section 5.1 mainly introduces the datasets which are used in this project. Section 5.2 discusses the data preprocessing strategies for image classification, object detection, and semantic segmentation tasks. Section 5.3 presents the hardware/software environment, and includes the introduction of network training tools. Section 5.4 introduces the training strategies, which are various based on the dataset. Finally, section 5.5 mainly describes the experimental results of MobileNetv2 and the improved network on different image tasks.

5.1 Datasets

In this project, we mainly focus on three different image tasks, image classification, object detection, and semantic segmentation. Three different tasks have related datasets to be used to train the network and compare the performance between different models. These datasets will be introduced by types of image tasks.

5.1.1 Image classification

There are three datasets for the image classification task, which are flowers dataset [12], Caltech-101 [13], and Caltech-256 [7]. The flowers dataset contains over 3k images and 5

(a) (b)

Figure 5.1: Two pictures from the Flowers dataset [12].

classes in total, all images in this dataset are flowers, and 5 categories represent 5 different types of flowers. Since the flowers dataset provides the image task of distinguishing different types of flowers, it tests the network capability of capturing fine-grained features, which are the features between similar objects. Moreover, the flowers in many images of the flowers dataset are not major objects, for example, the images of a person and flowers and images of a few flowers around a statue in Figure 5.1. From this aspect, the flowers dataset also tests the network capability of capturing fine-grained features.

Caltech-101 is the dataset which includes 101 categories and over 8k images. Caltech-101 provides general daily objects, such as airplanes, cups, etc. The images in Caltech-101 are generally pre-processed, including cropping or scaling images to make the target objects located in the major position of images. Another similar dataset we used is Caltech-256, which is the improved version of Caltech-101. The Caltech-256 have 257 categories and over 30k images. Caltech-256 provide more difficult tasks for network since there are some images which the target objects are not located in the major position, and some categories which are hard to recognize, such as frog. These two datasets are used to evaluate the network capabilities of capturing complex and rich features.

Figure 5.2: Random Horizontal Flipping.

5.1.2 Object Detection and Semantic Segmentation

Pascal VOC 2007 [5] was used as the object detection task dataset in this project. It includes objects from vehicles, households, animals, and others, which has 20 categories in total. The image labels for the object detection task in Pascal VOC 2007 include the object name and the bounding box coordinates, which include the coordinates of the bottom left corner and top right corner of the rectangular box.

For the semantic segmentation task, Pascal VOC 2012 [6] was used in this project. Pascal VOC 2012 is the improved version of Pascal VOC 2007, and the number of segmentation images is increased to almost 10k. The image labels for the segmentation task are formed as one-channel PNG files.

5.2 Data preprocessing

Data preprocessing is the necessary step for training networks. For the datasets which are used for the image classification task, 10% of images in each class will be split as the validation set, and the rest images will be used to train the network. All the datasets for image classification take the same data augmentations, which include random size cropping, random horizontal flipping, and normalization. The training images will be randomly resized to 224×224 during the training procedure of both the MobileNetv2 and the improved network, and each image has a 50% possibility of being horizontally flipped (as Figure 5.2).

The data augmentations used in the object detection task refer to the Pytorch implementation of Single Shot MultiBox Detector [3], which mainly contains random brightness, random contrast, random mirror, etc. The image size is 300×300 for the object detection task.

For the semantic segmentation task, data augmentations include random horizontal flipping (50% possibility) and random cropping. The images will be cropped to the size of 480×480.

5.3 Tools

The experiments are based on the Pytorch environment. One NVIDIA RTX 2060 is provided in this project as the GPU resource and the CUDA version is 11.7. The CPU which is AMD Ryzen 7 4800H is used in this project.

For network training tools, this project mainly uses a Python library called tqdm to visualize the training procedure in the console, and the Weights & Biases (also called wandb) to record the results of the segmentation training process in real-time. Additionally, matplotlib (which is a Python library) is used to visualize and analyse the data.

5.4 Training strategies

The loss function for image classification is the Cross-Entropy loss function. The AdamW [17] optimizer is applied in the training procedures related to all three image classification datasets. The networks are trained for 60 epochs when training on the flowers dataset, 120 epochs when training on Caltech-101, and 300 epochs when training on Caltech-256. The batch size of training on all three datasets is 16.

The learning rate schedulers are different depending on the dataset. For the flowers dataset and Caltech-101, the cosine learning rate scheduler is used. The learning rate is

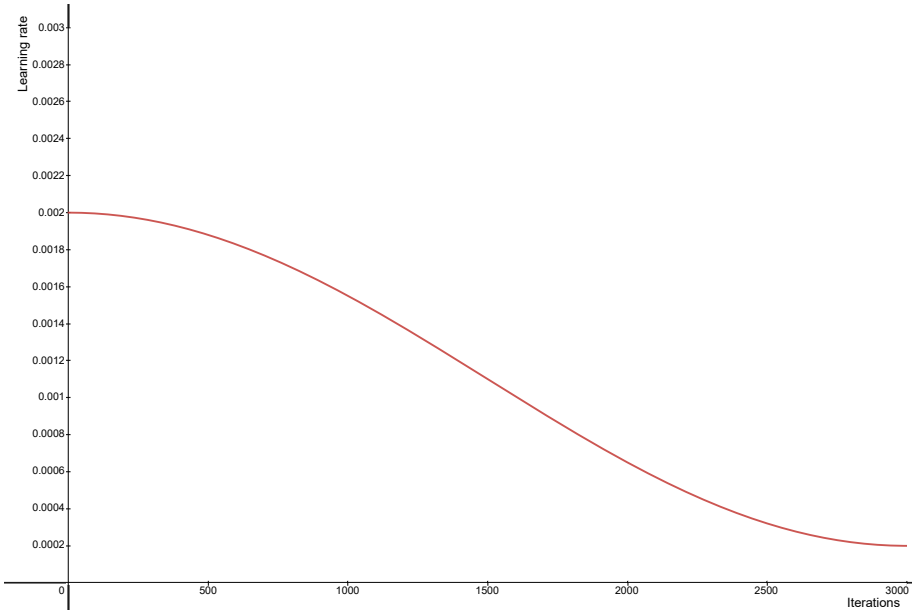


Figure 5.3: Cosine learning rate scheduler.

updated every iteration of training and follows the function below:

$$f(x) = f_{min} + (1 - f_{min})\frac{1 + cos(\frac{x \cdot \pi}{iterations})}{2}$$

$$lr(iteration) = lr_{initial} \cdot f(iteration)$$

Where f_{min} is the minimum value of the factor, $iterations$ is the total number of expectation iterations, $lr(iteration)$ and $lr_{initial}$ are the learning rate in the current iteration and the initial learning rate. Assuming the total training iterations are 3000, the initial learning rate is 0.002, and the minimum factor is 0.1, then the relationship between the learning rate and iteration can be described in Figure 5.3, during the training procedure, the learning rate will decrease from 0.002 to 0.0002.

Regarding training on the flowers dataset and Caltech-101, the initial learning rate is 0.002, and the minimum factor is 0.01, which means the range of learning rate is 0.002 to 0.00002.

Since Caltech-256 provide a more difficult classification task than the flowers dataset and Caltech-101, therefore, another learning rate scheduler called cosine annealing with warmup is considered. This scheduler renders the learning rate to gradually decrease

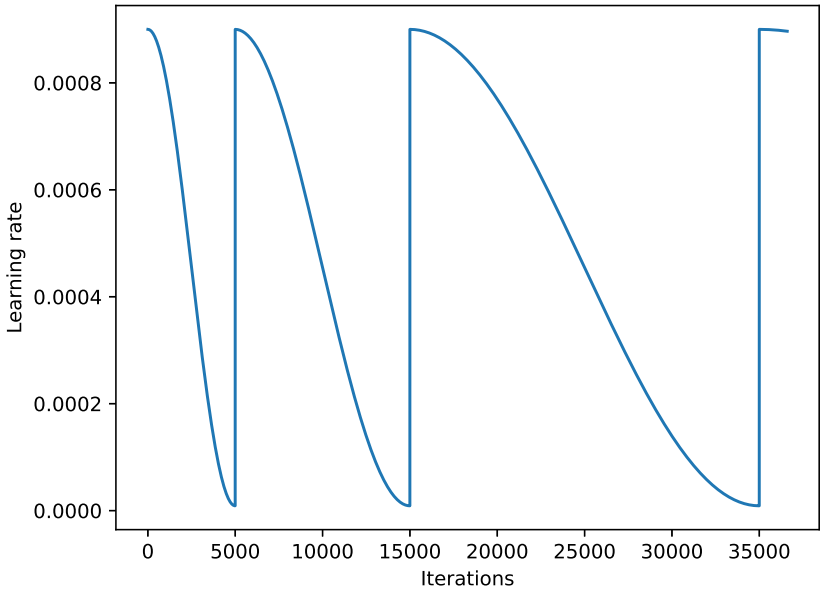


Figure 5.4: Cosine annealing learning rate scheduler with warmup. First iteration period is 5000, T_mul = 2, and learning rate between 0.0009 to 0.000009.

with the number of iterations according to the cosine curve. When the period is reached, the learning rate drops to its lowest point, and then immediately returns to the initial learning rate and repeats this process. This dynamic learning rate scheduler helps the model escape from local optima, and the details of how the learning rate changes by using this scheduler are shown in Figure 5.4. For training on Caltech-256, the initial learning rate is 0.002, the minimum learning rate is 0.00002, the first restart iteration period is 34560 iterations, and the T_mul, which is the factor of increasing the restart iterations, is 2.

For the Object Detection task, the networks are trained for 200 epochs without the pre-trained model. The images are resized to 300×300. The batch size is 24, and the initial learning rate is 0.01. The cosine annealing learning rate scheduler is used.

For semantic segmentation, considering training from scratch is too difficult for the semantic segmentation task, the pre-trained models are deployed to improve the results. The models (including all the models from both the MobileNetv2 and the improved network) trained on Caltech-256 are used as the pre-trained models. The image has been

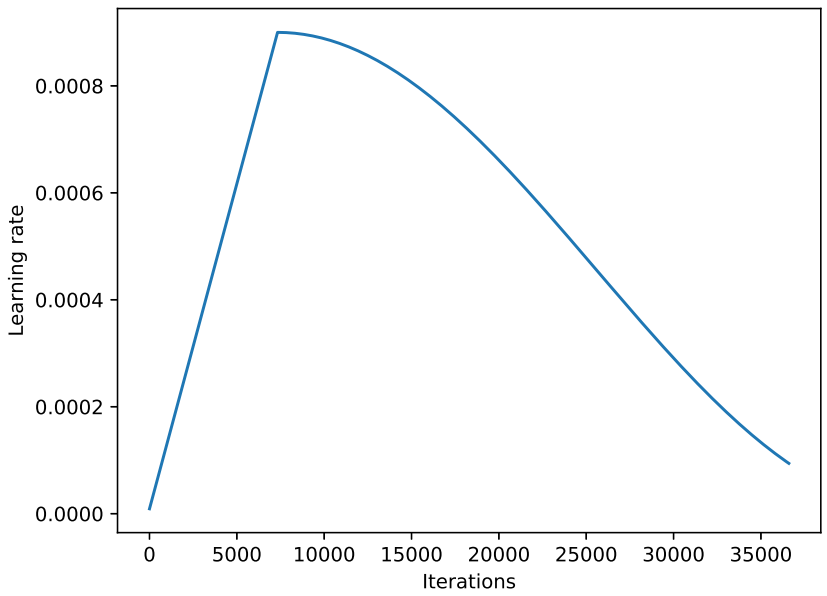


Figure 5.5: The learning rate scheduler of semantic segmentation.

adjusted to a size of 480×480, and the batch size is 8. The networks are trained for 200 epochs with pre-trained models. AdamW [17] is chose as the optimizer for this task. The learning rate ascended from 0.000009 to 0.0009 for the first 40 epochs, then generally cosine annealed to 0.000009. The specific change in the learning rate is shown in Figure 5.5.

5.5 Experiment results

5.5.1 Image classification

For the training results on the flowers dataset and Caltech-101, which are shown in the table 5.1, the improved model outperforms the original MobileNetv2 under relative expansion factors. In the comparison between the same expansion factors, the model trained on the flower dataset showed a minimum improvement of 0.3% and a maximum improvement of 1.7%, while the model trained on the Caltech-101 dataset showed an improvement between 0.4% to 1%. Meanwhile, comparing the model size data, it can be found that

under the same expansion factor, the improved model is smaller. These results demonstrate that introducing channel overlapping group convolution can effectively improve model performance while reducing model size, and this structure can be easily introduced into MobileNetv2 blocks. However, when we trained the models using the Caltech-256 dataset, we found that when the expansion factor was 4 and 6, the improved model showed a performance decrease compared to the original MobileNetv2 model. Specifically, from table 5.2, when the expansion factor was 4, the accuracy of the improved model decreased by 0.3% compared to the corresponding MobileNetv2 model, and when the expansion factor was 6, the accuracy decreased by 0.6%. In order to investigate the reasons for the performance degradation of the improved model on the Caltech-256 dataset, we exported the class accuracy of the trained model and the sample volume distribution data of Caltech-256. Then, we sorted the classes according to the size of the training sample size and mapped the class accuracy to the corresponding colours based on the colour bands ranging from 0 to 1. The relationship between the volume of training samples and the class accuracy is shown in the Figure A.2 and Figure A.3. Based on the results from the figures, we came up with an assumption. To prove this assumption, we reform a new sub dataset of the Caltech-256, and then, used this sub dataset to compare the performance of the two models again. The training results are shown in the table 5.3, which shows the performances of improved models trained on this sub dataset are better than MobileNetv2, this confirms our hypothesis. The details of our assumptions and the sub dataset will be discussed in the section 6.1.

	Top-1 accuracy (Flowers)	Model size (KB)	Top-1 accuracy (Caltech-101)	Model size (KB)
MobileNets_e2	86.30%	**3,717**	83.80%	**4,176**
MobileNets_e4	**86.80%**	5,617	**85.40%**	6,083
MobileNets_e6	85.70%	7,518	84.90%	7,983
MobileNetv2_e2	84.60%	4,180	82.80%	4,665
MobileNetv2_e4	86.00%	6,570	84.60%	7,055
MobileNetv2_e6	85.20%	8,960	84.50%	9,445

Table 5.1: The Top-1 accuracy of improved network (MobileNets) and MobileNetv2 on the flowers dataset and caltech-101. e2, e4, e6 represent the expansion factor equals 2, 4, and 6.

	Top-1 accuracy	Model size (KB)	Parameters
MobileNets_e2	69.70%	**4,985**	**2,175,528**
MobileNets_e4	71.40%	6,885	2,652,520
MobileNets_e6	71.30%	8,785	3,129,512
MobileNetv2_e2	67.10%	5,447	2,300,264
MobileNetv2_e4	71.70%	7,837	2,902,568
MobileNetv2_e6	**71.90%**	10,227	3,504,872

Table 5.2: The Top-1 accuracy of improved network and MobileNetv2 on Caltech-256.

	MobileNets	MobileNetv2
e4	**81.5%**	81.1%
e6	**81.9%**	80.7%

Table 5.3: The Top-1 accuracy of improved network and MobileNetv2 on sub dataset of Caltech-256 when the expansion factor equals 4 and 6.

5.5.2 Object Detection

The results of Object Detection task training on Pascal VOC 2007 are shown in the table 5.4. For the expansion factor equalling 2, the mean average precision (mAP) of the improved model is over 5% higher than the same level of the MobileNetv2 model. For the other two model sizes, the improved models get 2.5% (expansion factor = 4) and 1.5% (expansion factor = 6) mAP improvement.

	mAP	Inference time (ms)/ per image
MobileNets_e2	41.5	6.08
MobileNets_e4	45.6	6.25
MobileNets_e6	**46.5**	6.83
MobileNetv2_e2	35.9	**5.77**
MobileNetv2_e4	43.1	5.93
MobileNetv2_e6	45	6.06

Table 5.4: The mean average precision and inference time of improved network and MobileNetv2 on Pascal VOC 2007.

5.5.3 Semantic Segmentation

For the Semantic Segmentation task, the backbone models were pre-trained on Caltech-256 for 300 epochs, then, training the entire networks on the segmentation dataset for 200 epochs, the results are shown in table 5.5. Similar to the results of the Object Detection task, when the expansion factor equals 2, there was over 5% mean Intersection over Union (mIoU) improvement, and other model sizes also achieved 0.3% (expansion factor = 4) and 0.9% (expansion factor = 6) improvement comparing with MobileNetv2 models. The training curves of all models are compared in Figure A.1.

	meanIoU
MobileNets_e2	45.7
MobileNets_e4	46
MobileNets_e6	**47.4**
MobileNetv2_e2	40.6
MobileNetv2_e4	45.7
MobileNetv2_e6	46.5

Table 5.5: The mean intersection over union of improved network and MobileNetv2 on Pascal VOC 2012.

Chapter 6

Discussion

This section will mainly analyse the results of the experiments, and discuss the reason for the unexpected results to further explore the properties of the improved network.

6.1 Image classification

From the experiment results in the table 5.1, it is clear that, on the flowers dataset and Caltech-101, the improved models achieve better performance than models of MobileNetv2. This proves that introducing channel-overlapping group convolution can improve the performance of the MobileNetv2 block. However, for the experiment results from the dataset of Caltech-256, although the improved models perform better on the level of expansion factor equalling 2, they perform slightly worse than the MobileNetv2 when the expansion factor equals 4 and 6. Based on this experimental phenomenon, a question has been raised, which is why the improved model only performs worse than MobileNetv2 on the Caltech-256 dataset. To explore the reason for this phenomenon, we compared the difference between the Caltehc-101 and the Caltech-256, since the Caltech-256 is the improved version of the Caltech-101, there is not much difference from the data content aspect. The major difference is the volume of samples and number of categories, specifically, the Caltech-256 contains 257 categories which is dramatically larger than the Caltech-101. After further exporting the training sample volume for each class

in Caltech-256 and sorting it in descending order of data size, we obtained the sample volume distribution for this dataset, which is shown in Figure 6.1. From the figure, it is obvious that the sample volume follows long-tail distribution, which means the few categories contain a considerable volume of training samples. Then we output the accuracy of each class of the model and map them to colours, as shown in the Figure A.2 and Figure A.3. The blue to red colour band represents the accuracy from 1 to 0. When we reflect the classes accuracy on the sample volume distribution map, we found that, for both MobileNetv2 and the improved network, the bad-performing classes are mainly concentrated in the tail area. Based on this, we made an assumption to explain, which is the channel-overlapping group convolution may be sensitive to the volume of training samples. Specifically, when the distribution of training samples follows the long-tail distribution or the dataset cannot provide sufficient training data, then the improvement from channel-overlapping group convolution will decrease, and if the tail area is too long, then the performance of the network introduced this structure may be worse than original network. To prove our assumption, we decided to take the top 20% of categories which contain the highest data volume, then, form them as a new sub dataset. This new sub dataset contains 53 categories which occupy a considerable volume of training samples. Then we trained MobileNetv2 and the improved network on this new sub dataset using the similar training strategy as when training on Caltech-101. The training results are shown in the table 5.3, from the results, the improved network, for both expansion factors equalling 4 and 6, achieved better results than MobileNetv2 models. The results prove our assumption, which is when the dataset provides excessive categories and insufficient training data simultaneously, introducing channel-overlapping group convolution in the network may lead to the performance dropping. On the other hand, if the dataset can provide sufficient training data, then the network introduced channel-overlapping group convolution will improve the network performance.

Comparing the performances under different expansion factors, we found that, for both the improved network and MobileNetv2, when the expansion factor equals 6, the

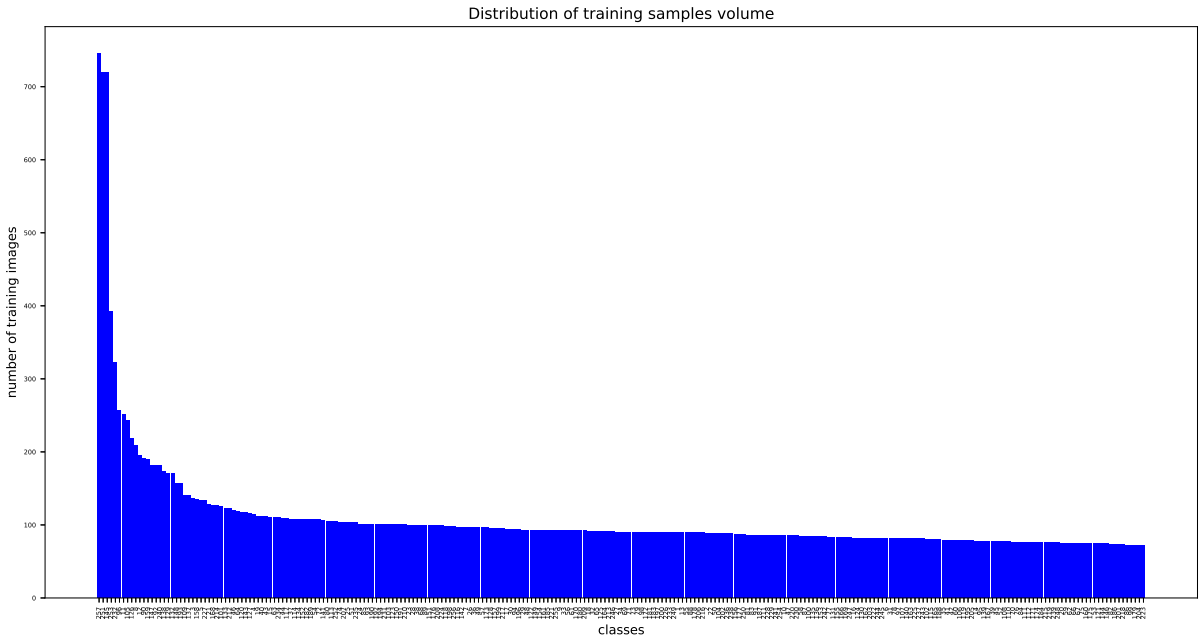


Figure 6.1: Training samples distribution of Caltech-256.

performance of networks will drop. In this concern, increasing the expansion factor may not be a good way to increase the performance of the network, in fact, since increasing the expansion factor will ascend the model size, therefore, blindly increasing the expansion factor can severely damage the efficiency of the network. From the experiment results, when the expansion factor equals 2, the improved network gained the highest performance improvement on all three datasets. Additionally, considering the guideline from ShuffleNetv2 [18], the MAC reaches the lowest bound when the expansion factor equals 2 since the input channel and output channel are equal for each group convolutional filter. Therefore, when the expansion factor is 2, the improved network enjoys the highest efficiency, and the performance improvement will be decreased as the expansion factor increases. From the results of experiments, it is not recommended to set the expansion factor higher than 4.

By comparing the model sizes of the improved network and MobileNetv2 from the table 5.1, this structure can effectively decrease the volume of parameters. For a single

original pointwise convolution layer, the parameters volume can be calculated by:

$$Parameters_{pw} = C_i C_o$$

Where C_i indicates the number of input channels and C_o indicates the number of output channels. After introducing the channel-overlapping group convolution, which the number of groups is four and there is one overlapping group between two neighbour groups, the parameter volume of the improved pointwise layer can be calculated by the following formula:

$$Parameters_{ipw} = 4 \times 2\frac{C_i}{4} \times \frac{C_o}{4} = \frac{C_i C_o}{2}$$

Which has 4 convolutional filters (4 groups), and the input channel for each group is 2 $\frac{C_i}{4}$ and the output channel is $\frac{C_o}{4}$. Therefore, introducing channel-overlapping group convolution can solidly decrease the volume of parameters, which is the reason why improved networks enjoy the smaller model size than MobileNetv2.

6.2 Object Detection

According to the results shown in the table 5.4, the improved network achieved better performance on the object detection task than MobileNetv2. When the expansion factor equals 2, then the improved network gained the highest performance improvement, which followed our expectation, however, the inference time of improved models is generally slower than MobileNetv2. According to the first guideline from ShuffleNetv2 [18], this structure can decrease the MAC, since for the expansion procedure of MobileNetv2 block, the ratio of input channel to output channel is the value of the expansion factor, which:

$$\frac{C_o}{C_i} = expansion factor$$

After introducing the channel-overlapping group, the ratio for each group convolution filter will be:

$$\frac{\frac{C_o}{4}}{\frac{2C_i}{4}} = \frac{expansion factor}{2}$$

The expansion factor is larger than 1, therefore, introducing this structure will decrease the MAC, the inference speed of the network should be faster, so, why does the network spend more inference time than the original structure? This is because, channel-overlapping group convolution is still the group convolution, according to the third guideline of ShuffleNetv2, this will introduce extra overheads, for example, kernel launching and synchronization [18]. Based on this, the slower inference speed is because the inference time saved by reducing memory access costs can only partially alleviate the time gap caused by the overheads introduced by group convolution under current hardware conditions, but cannot fully compensate for the lost time. However, it is acceptable that, based on the experimental results (table 5.4) of object detection, the inference time difference between the improved network and MobileNetv2 is within 1 millisecond per image. There is one thing we want to further explain, according to the second guideline, the group convolution will increase the MAC, this guideline has a precondition, which is under similar FLOPs. Since introducing channel-overlapping convolution decreases the FLOPs, it does not fit the precondition of the second guideline. Assuming the input feature map size is hw, then the MAC of the original pointwise convolution can be described as:

$$MAC_{pw} = hw(C_i + C_o) + C_iC_o$$

After introducing channel-overlapping group convolution, the MAC of improved pointwise layer is:

$$MAC_{ipw} = hw(C_i + C_o) + \frac{C_iC_o}{2}$$

Clearly, the MAC is decreased after introducing our structure.

6.3 Semantic Segmentation

From the table 5.5, the improved models achieved better results than MobileNetv2, especially when the expansion factor is equal to 2, the mIoU of the improved model is over 5% higher than MobileNetv2. The excellent performance proved that this structure can effectively improve the performance of the network on the Semantic Segmentation task. The relationship curve between mIoU and epoch is shown in the Figure A.1.

Chapter 7

Conclusion

From both theoretical analysis and experimental results, the channel-overlapping group convolution can effectively improve the efficiency of the network. First, by introducing the channel-overlapping group convolution, the volume of parameters and model size get solid optimization. Additionally, the computational cost and FLOPs are also decreased. Second, generally, the network performances will be lifted after introducing our structure, which further improves the efficiency of the network. Based on the results of the experiment, the expansion factor can affect the improvement of the network. Specifically, when the expansion factor is equal to 2, introducing channel overlapping group convolution results in the highest performance improvement. The performance improvement decreases with the increase of the expansion factor. From the results, generally, it is not recommended to set the expansion factor greater than 4. Third, this structure also contains high portability, since this structure can be easily implemented in the MobileNetv2 block, and this improved block can replace the original MobileNetv2 block in other models, such as MobileViT [19]. In this project, we fixed the number of groups to be 4, and one overlapping group between neighbour groups. This is the best setting for MobileNetv2 and also is the setting we recommend, however, based on different networks, this setting may not be perfect, the setting could be changed to reach the balance point between performance and parameter quantity or fit the network structure.

However, this structure also has some drawbacks. When the volume of samples in each

class in the dataset follows a long-tail distribution, the performance improvement brought by introducing channel-overlapping group convolution may be reduced. If the tail region is excessively long, performance degradation may occur. Additionally, due to the overheads introduced by group convolution, although channel-overlapping group convolution can effectively reduce MAC, the inference time saved is insufficient to compensate for the additional inference time caused by network fragmentation, resulting in a slight decrease in the inference speed of the model.

Based on the advantages and disadvantages of this structure, channel-overlapping group convolution can effectively reduce model size and improve model performance as long as it meets some preconditions. When introducing this structure into MobileNetv2, the smaller the number of parameters, the more significant the model improvement. The two preconditions that need to be met are: first, the sample volume distribution of the dataset should not follow the long-tail distribution with the tail area which is too long, and second, the model parameters should be set reasonably. For this project, the excessive number of groups and expansion factor can seriously damage model performance. Setting the number of groups to 4 and the expansion factor to 2 and 4 is the most reasonable choice.

Due to limited hardware resources, we were unable to test the performance of the improved network on ImageNet-1k [21] and the performance of the improved inverted residual block on other models, such as MobileViT. In addition, we only explored the optimization effect of channel overlapping group convolution on pointwise convolution, and the optimization effect of this structure on standard convolution is still unknown. We hope to have the opportunity to further explore this structure in the future research.

List of References

[1] Chen, L.C., Papandreou, G., Schroff, F. et al. (2017) Rethinking atrous convolution for semantic image segmentation. **arXiv preprint arXiv:1706.05587**.

[2] Cubuk, E.D., Zoph, B., Shlens, J. et al. (2020) "Randaugment: Practical automated data augmentation with a reduced search space." In **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops**. pp. 702–703

[3] deGroot, M. and Brown, E. (2019) Ssd: Single shot multibox object detector, in pytorch. Github. https://github.com/amdegroot/ssd.pytorch.

[4] Dosovitskiy, A., Beyer, L., Kolesnikov, A. et al. (2020) An image is worth 16x16 words: Transformers for image recognition at scale. **arXiv preprint arXiv:2010.11929**.

[5] Everingham, M., Van Gool, L., Williams, C.K.I. et al. (2007) The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[6] Everingham, M., Van Gool, L., Williams, C.K.I. et al. (2012) The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[7] Griffin, G., Holub, A. and Perona, P. (2022) Caltech 256.

[8] He, K., Zhang, X., Ren, S. et al. (2016) "Deep residual learning for image recognition." In **Proceedings of the IEEE conference on computer vision and pattern recognition**. pp. 770–778

[9] Howard, A., Sandler, M., Chu, G. et al. (2019) "Searching for mobilenetv3." In **Proceedings of the IEEE/CVF international conference on computer vision**. pp. 1314–1324

[10] Howard, A.G., Zhu, M., Chen, B. et al. (2017) Mobilenets: Efficient convolutional neural networks for mobile vision applications. **arXiv preprint arXiv:1704.04861**.

[11] Hu, J., Shen, L. and Sun, G. (2018) "Squeeze-and-excitation networks." In **Proceedings of the IEEE conference on computer vision and pattern recognition**. pp. 7132–7141

[12] Kaggle (2021) Flowers dateset. https://www.kaggle.com/datasets/imsparsh/flowers-dataset. Accessed on 2023-6-20.

[13] Li, F.F., Andreeto, M., Ranzato, M. et al. (2022) Caltech 101.

[14] Liu, W., Anguelov, D., Erhan, D. et al. (2016) "Ssd: Single shot multibox detector." In **Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14**. Springer. pp. 21–37

[15] Liu, X., Peng, H., Zheng, N. et al. (2023) "Efficientvit: Memory efficient vision transformer with cascaded group attention." In **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. pp. 14420–14430

[16] Liu, Z., Mao, H., Wu, C.Y. et al. (2022) "A convnet for the 2020s." In **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition**. pp. 11976–11986

[17] Loshchilov, I. and Hutter, F. (2017) Decoupled weight decay regularization. **arXiv preprint arXiv:1711.05101**.

[18] Ma, N., Zhang, X., Zheng, H.T. et al. (2018) "Shufflenet v2: Practical guidelines for efficient cnn architecture design." In **Proceedings of the European conference on computer vision (ECCV)**. pp. 116–131

[19] Mehta, S. and Rastegari, M. (2021) Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. **arXiv preprint arXiv:2110.02178**.

[20] Mehta, S. and Rastegari, M. (2022) Separable self-attention for mobile vision transformers. **arXiv preprint arXiv:2206.02680**.

[21] Russakovsky, O., Deng, J., Su, H. et al. (2015) Imagenet large scale visual recognition challenge. **International journal of computer vision**, 115: 211–252

[22] Sandler, M., Howard, A., Zhu, M. et al. (2018) "Mobilenetv2: Inverted residuals and linear bottlenecks." In **Proceedings of the IEEE conference on computer vision and pattern recognition**. pp. 4510–4520

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

[24] Tan, M. and Le, Q. (2019) "Efficientnet: Rethinking model scaling for convolutional neural networks." In **International conference on machine learning**. PMLR. pp. 6105–6114

[25] Wadekar, S.N. and Chaurasia, A. (2022) Mobilevitv3: Mobile-friendly vision transformer with simple and effective fusion of local, global and input features. **arXiv preprint arXiv:2209.15159**.

[26] Zhang, H., Cisse, M., Dauphin, Y.N. et al. (2017) mixup: Beyond empirical risk minimization. **arXiv preprint arXiv:1710.09412**.

[27] Zhang, X., Zhou, X., Lin, M. et al. (2018) "Shufflenet: An extremely efficient convolutional neural network for mobile devices." In **Proceedings of the IEEE conference on computer vision and pattern recognition**. pp. 6848–6856

[28] Zhong, Z., Zheng, L., Kang, G. et al. (2020) "Random erasing data augmentation." In **Proceedings of the AAAI conference on artificial intelligence**. pp. 13001–13008

APPENDIX

A.1 Git repository

The code for this project has been uploaded to the GitLab repository, which can be accessed through the following link:

https://git.cs.bham.ac.uk/projects-2022-23/hxl226.

There are three main folders in the repository: classification, object detection, and semantic segmentation. These three folders represent the three image tasks in this project.

The classification folder contain 4 folders and 3 files. 4 folders contain the model files, log files, and results files. Each folder contain:

- first_laryer_compar: the model files, log files, and result files of table 4.2.
- results_in_report: contain the files of classification experiments of improved network and MobileNetv2 on the flowers dataset and Caltech-256, the results of Caltech-101 are in the 'first_laryer_compar' folder.
- sub-dataset-results: result files of sub dataset experiment in table 5.3.
- two-overlapping-groups: result files of table 4.1.

The three files are:

- model_s: the network structure code file of improved network.
- model_v2: the network structure code file of MobileNetv2.
- train: training script for classification tasks

The Segmentation folder contains the code related to the semantic segmentation task. The segmentation training code references this GitHub repository. My contribution in

this part is the 'model_s.pth' file in the src folder, and I modified some code files to allow improved model as the backbone of deeplabv3. The 'results' folder contains the experiment (table 5.5) results files.

The Object Detection folder is based on this GitHub repository. I created 'mobilenet_-s_ssd_lite.py' file in the path of '/hxl226/Object Detection/vision/ssd' to create an interface with the improved models. The improved network file 'model_s.py' is in the path of '/hxl226/Object Detection/vision/nn'. The 'results' folder contain the result files of segmentation task (the model files are too big to upload). When using the segmentation project, if you want to run our model, please ensure that lines 68 and 69 of the code in the ssd.py file are available, if you want to run the other models, like MobileNetv2, then please dis able this two lines. This two lines of code are:

```
x_a, x_b, x_c, x_d = x.chunk(4, dim=1)
x = torch.cat((x_a, x_b, x_b, x_c, x_c, x_d, x_d, x_a), dim=1)
```

The configuration when we executing 'traing_ssd.py' is:

```
--validation_dataset
./VOCtest_06-Nov-2007/VOCdevkit/VOC2007/
--net
mb2-ssd-lite
--batch_size
24
--num_epochs
200
--scheduler
cosine
--lr
```

```
0.01
--t_max
200
--num_workers
0
```

The torch version of this project is 1.13.1, cuda version is 11.7. The hardware resource information is in section 5.3.

A.2 Semantic Segmentation training curve

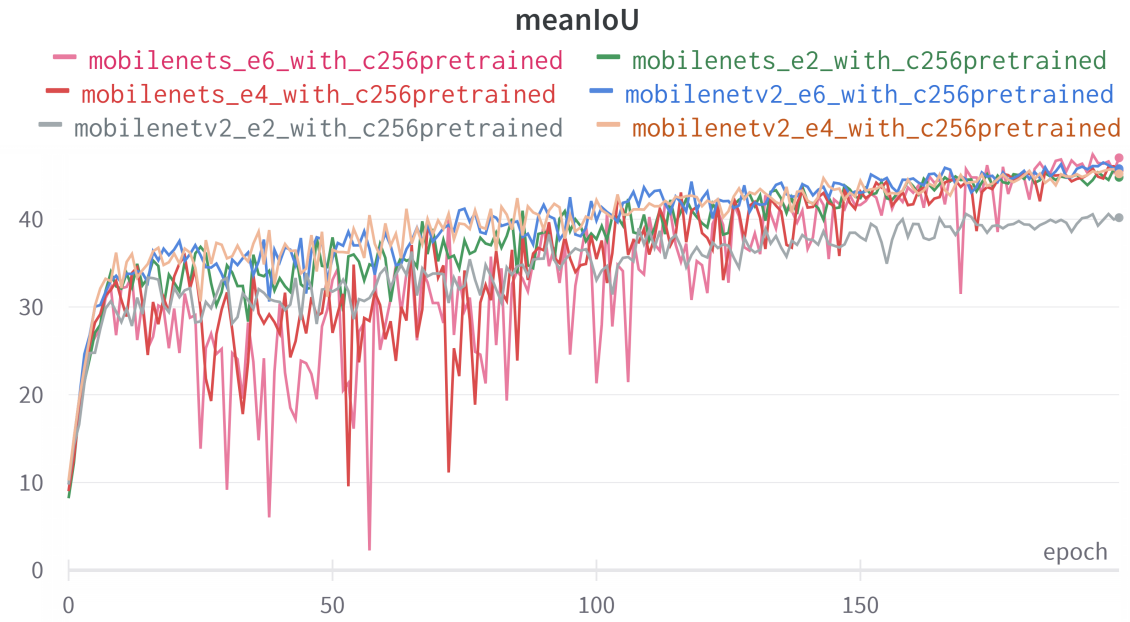
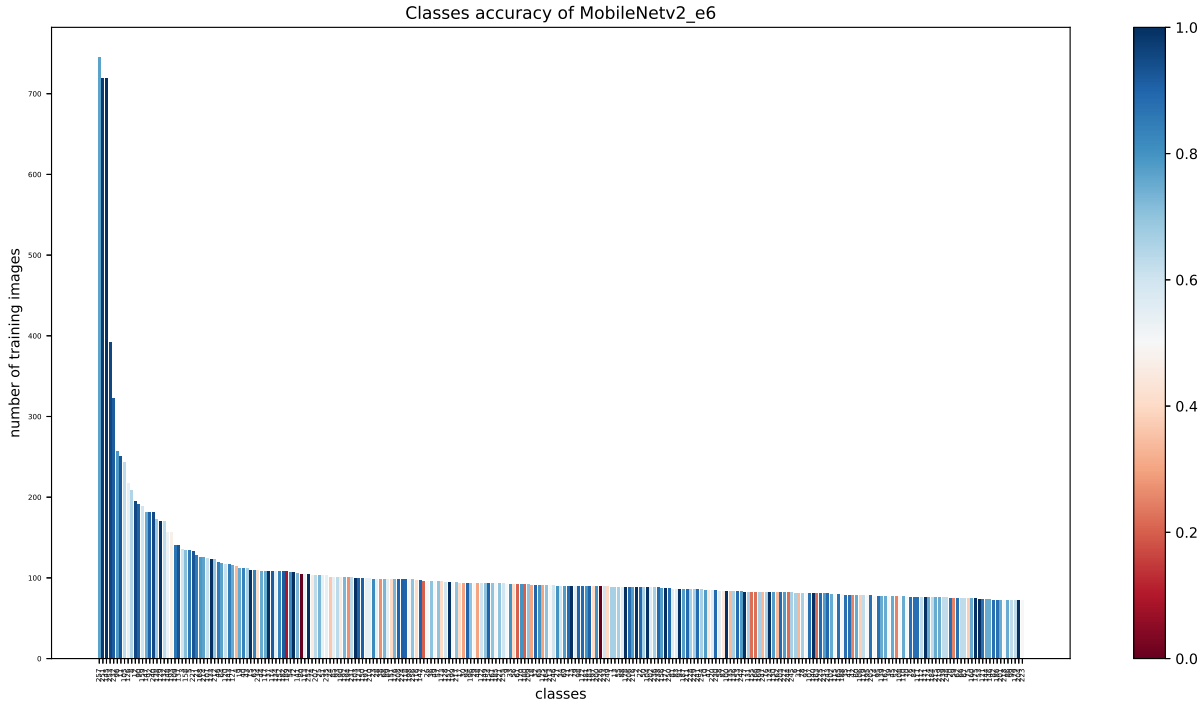
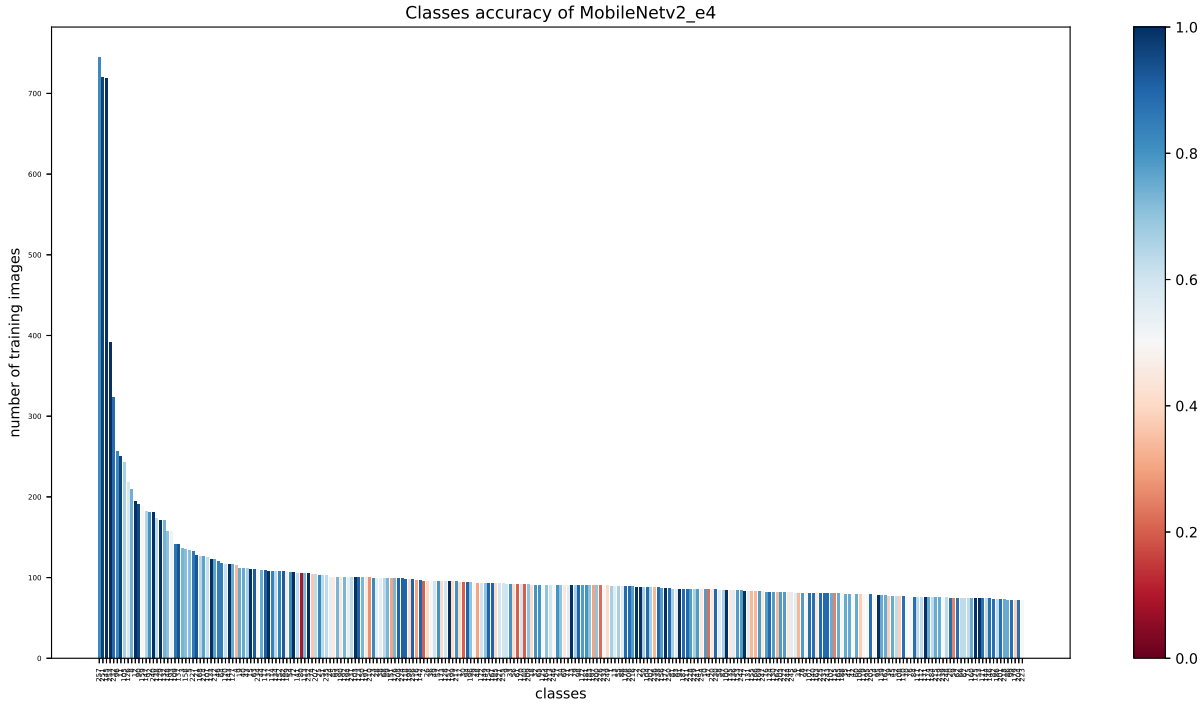


Figure A.1: The comparison of training procedure.

A.3 Classes accuracy diagrams

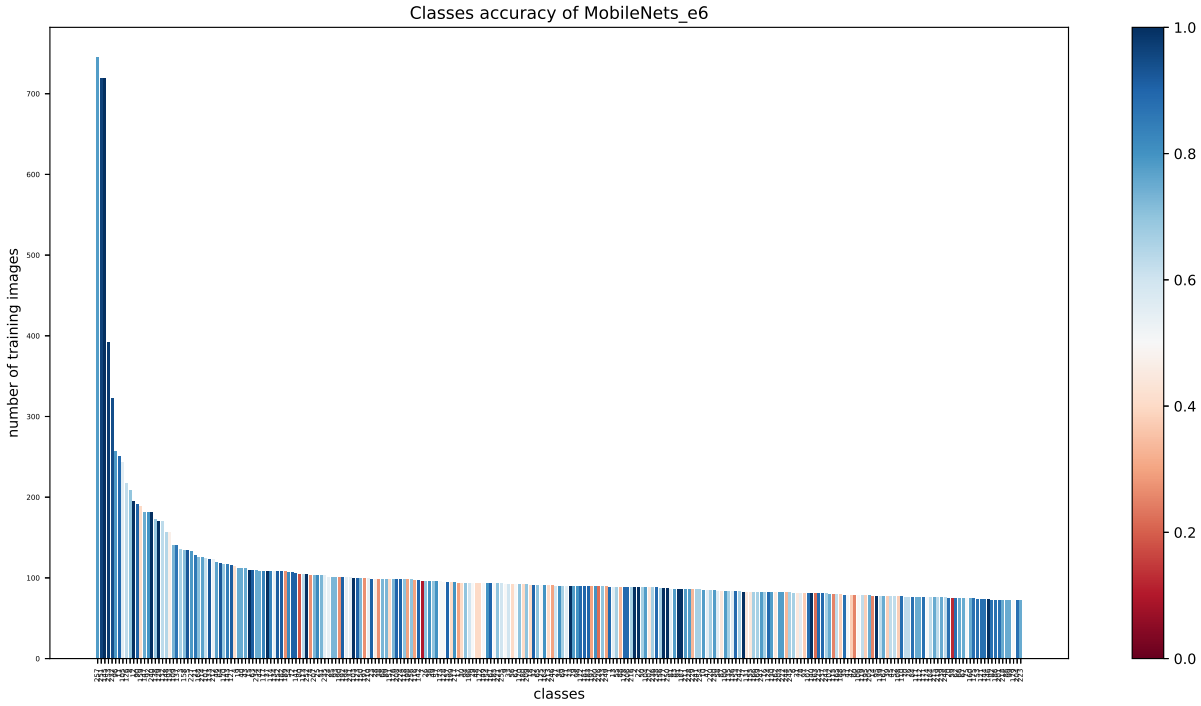


(a)

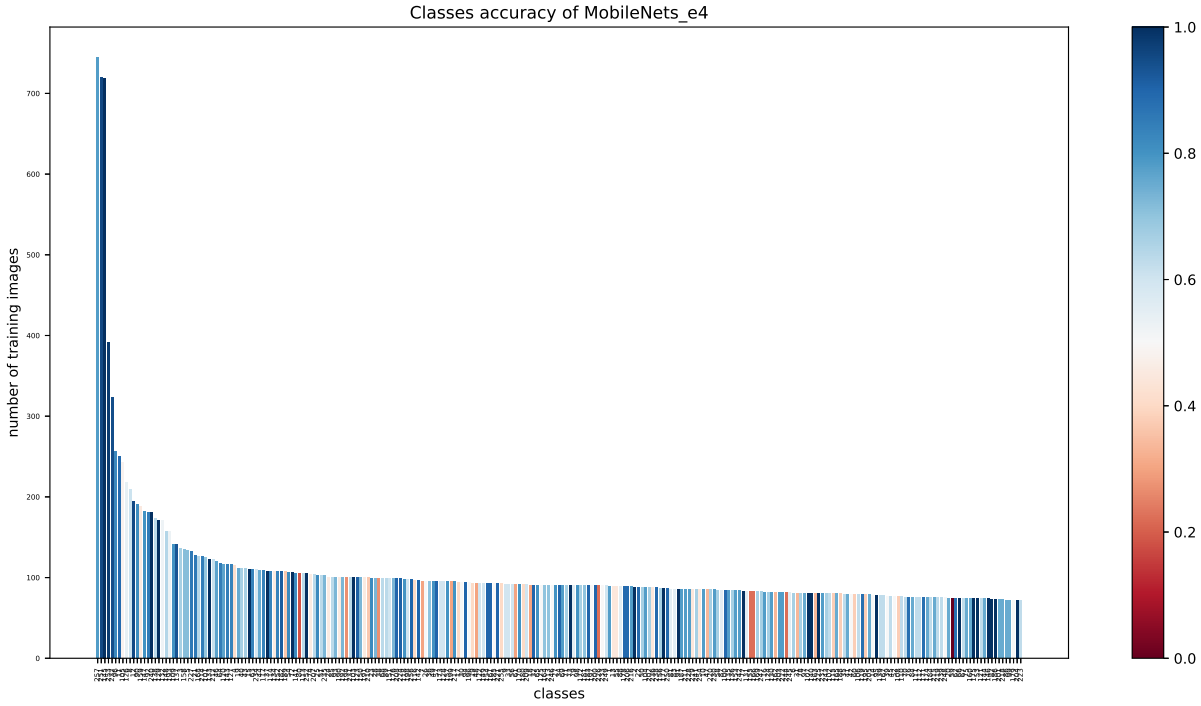


(b)

Figure A.2: Classes accuracy of Mobilenetv2 when expansion factor is 4 and 6.



(a)



(b)

Figure A.3: Classes accuracy of improved network when expansion factor is 4 and 6.