The Enhanced Hybrid MobileNet

Hong-Yen Chen
Department of Electrical Engineering
National Taiwan Normal University
Taipei, Taiwan
40475009H@ntnu.edu.tw

Chung-Yen Su *
Department of Electrical Engineering
National Taiwan Normal University
Taipei, Taiwan
scy@ntnu.edu.tw

***Abstract*—Although complicated and deep neural network models can achieve high accuracy of image recognition, they require huge amount of computations and parameters and are not suitable for mobile and embedded devices. As a result, MobileNet [1] was proposed, which can reduce the amount of parameters and computational cost dramatically. MobileNet [1] is based on depthwise separable convolution and has two hyper-parameters width multiplier and resolution multiplier which can trade-off between accuracy and latency. In this paper, we propose two different methods to improve MobileNet [1], which are based on adjusting two hyper-parameters width multiplier and depth multiplier, combing max pooling or Fractional Max Pooling [2] with MobileNet [1]. We tested the improved models on images classification database CIFAR with good results[1].**

Keywords—deep learning; MobileNet; neural networks

I. INTRODUCTION

Since AlexNet [3] achieved ImageNet's Champion [4], the general trend has been to make more complicated and deeper networks in order to meet higher accuracy [5, 6, 7, 8]. Until recently, people began to modify models using less parameters and computational cost without dropping the accuracy too much [9, 10 , 11 , 12]. Thus, MobileNet [1] is emerged. It provided a solution for mobile and embedded devices. MobileNet [1] use a special convolution named depthwise separable convolution instead of standard convolution because of the depthwise separable convolution using between 8 to 9 times less computational cost than standard convolution and only decreasing little accuracy. There are two hyper-parameters, α named width multiplier, and ρ named resolution multiplier. By adjusting α and ρ in the MobileNet [1] can trade-off between computations and accuracy. However, both α and ρ reduce the computational cost or parameters of models by sacrificing the accuracy. But if the accuracy of real-time applications is too low will be useless. So we propose two methods which are adjusting a hyper-parameter δ called depth multiplier and adding various kinds of pooling to MobileNet [1] to increase accuracy.

In this paper, we will propose two different methods to increase the accuracy of MobileNet [1] and also use a hyper-parameter α mentioned in MobileNet [1] to reduce the computational cost and parameters at the same time. The paper is organized as follows. Section 2 introduces the prior works about structure of MobileNet [1] and various pooling patterns. Section 3 describes the main idea of the proposed methods about using a hyper-parameter δ and adding two different kinds of max pooling in MobileNet [1]. Section 4 shows experimental results on CIFAR-10 and CIFAR-100. Section 5 gives a concluding remark.

II. PRIOR WORKS

Depthwise separable convolution, which is a form of factorized convolution, is the core layer in MobileNet [1], and the characteristic of it is splitting standard convolutions into a depthwise convolution and a 1×1 convolution named pointwise convolution. In the MobileNet [1], the depthwise convolution means that each of filters only uses single input channel to do convolution. Then the pointwise convolution layer integrates the depthwise convolution layer output and does 1×1 convolution. Depthwise separable convolution uses between 8 to 9 times less computational cost than standard convolution. Although MobileNet [1] structure is already small and low latency, MobileNet [1] also proposed two hyper-parameters, α named width multiplier, which means the ratio of amount of filters, and ρ named resolution multiplier, which means the ratio of the resolution of image. By adjusting α and ρ can reduce the amount of parameters or computational cost with decreasing the accuracy.

There are various kinds of pooling using in neural network models. Max Pooling Kernel size 2×2 Stride 2 is the common choice for building convolutional networks because it keeps most of the features and quickly reduces the size of the hidden layers. But according to [3], the effect of 3×3 pooling regions overlapping with stride 2 is greater than the common pooling. Different from the general pooling which can only downsample the images in integer multiple, Fractional Max Pooling [2] is able to downsample the images in decimals and able to increase the accuracy of networks.

III. THE PROPOSED METHODS

In this section, we will illustrate how we applied a special hyper-parameter δ named depth multiplier and the techniques mentioned in section 2 on MobileNet [1].

[1] This work is partly supported by Ministry of Science and Technology, R.O.C. under Contract No. MOST 106-2221-E-003-011.

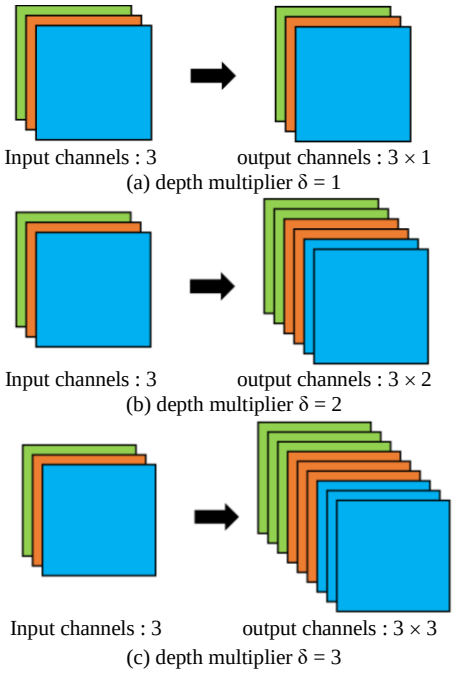

Figure 1. The operating principles of depth multiplier δ.

A. *MobileNet architecture in different α and δ*

Because one input channel may with more than one feature can be extracted in the depthwise convolution, we propose an approach which is through adjusting a hyper-parameter δ named depth multiplier in the depthwise convolution to change the number of feature maps which one input channel produces. δ is an integer, typically values are 1, 2 and 4, meaning the number of depthwise convolution output channels for each input channel and δ = 1 is the baseline in MobileNet [1]. For example, if input channels is equal to 3 and δ is equal to 2, each of input channels can be extracted two feature maps and the output channels number is equal to 6. The detailed operating principle of δ is shown in Figure 1. Then we found that the computational cost and amount of parameters will be increased if δ raised. So we also adjusted width multiplier α at the same time to reduce the computational cost and amount of parameters, the reason why we didn't adjust ρ to reduce the computational cost of models is if we want to use the MobileNet [1] to do object detection, reducing the resolution of the image may drastically lose the detection effect. By adjusting two hyper-parameters, respectively α and δ, we can change the amount of parameters and computational cost in the MobileNet [1].

The amount of parameters of a depthwise convolution is:

$$D_K \cdot D_K \cdot (\alpha M \cdot \delta) \quad (1)$$

The amount of parameters of a pointwise convolution is:

$$(\alpha M \cdot \delta) \cdot \alpha N \quad (2)$$

The computational cost of a depthwise convolution is:

$$D_K \cdot D_K \cdot (\alpha M \cdot \delta) \cdot D_F \cdot D_F \quad (3)$$

The computational cost of a pointwise convolution is:

$$(\alpha M \cdot \delta) \cdot \alpha N \cdot D_F \cdot D_F \quad (4)$$

The meaning of each notation are the number of output channels N, the number of input channels M, the kernel size of filters D_K and the feature maps size D_F. The model structure is shown in **Table 1**.

Table 1. MobileNet in different α and δ architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	**3 × 3 × 3 × 32α**	**32 × 32 × 3**
Conv dw / s1	**3 × 3 × (32α× δ)**	**16 × 16 × 32α**
Conv / s1	**1 × 1 × (32α× δ) × 64α**	**16 × 16 × (32α× δ)**
Conv dw / s2	**3 × 3 × (64α× δ)**	**16 × 16 × 64α**
Conv / s1	**1 × 1 × (64α× δ) × 128α**	**8 × 8 × (64α× δ)**
Conv dw / s1	**3 × 3 × (128α× δ)**	**8 × 8 × 128α**
Conv / s1	**1 × 1 × (128α× δ) × 128α**	**8 × 8 × (128α× δ)**
Conv dw / s2	**3 × 3 × (128α× δ)**	**8 × 8 × 128α**
Conv / s1	**1 × 1 × (128α× δ) × 256α**	**4 × 4 × (128α× δ)**
Conv dw / s1	**3 × 3 × (256α× δ)**	**4 × 4 × 256α**
Conv / s1	**1 × 1 × (256α× δ) × 256α**	**4 × 4 × (256α× δ)**
Conv dw / s2	**3 × 3 × (256α× δ)**	**4 × 4 × 256α**
Conv / s1	**1 × 1 × (256α× δ) × 512α**	**2 × 2 × (256α× δ)**
5 ×		
Conv dw / s1	**3 × 3 × (512α× δ)**	**2 × 2 × 512α**
Conv / s1	**1 × 1 × (512α× δ) × 512α**	**2 × 2 × (512α× δ)**
Conv dw / s2	**3 × 3 × (512α× δ)**	**2 × 2 × 512α**
Conv / s1	**1 × 1 × (512α× δ) × 1024α**	**1 × 1 × (512α× δ)**
Conv dw / s1	**3 × 3 × (1024α× δ)**	**1 × 1 × 1024α**
Conv / s1	**1 × 1 × (1024α× δ) × 1024α**	**1 × 1 × (1024α× δ)**
Avg Pool / s1	**Global average pooling**	**1 × 1 × 1024α**
FC / s1	**1 × 1 × 1024α× classnumber**	**1 × 1 × 1024α**
Softmax / s1	**Classifier**	**1× 1 × classnum**

We can calculate the ratio of the amount of parameters (a) and computational cost (b) between the modified depthwise separable convolution and the original depthwise separable convolution i.e. [α= 1 and δ = 1] .

(a) The ratio of amount of parameters:

$$\frac{D_K \cdot D_K \cdot (\alpha M \cdot \delta) + (\alpha M \cdot \delta) \cdot \alpha N}{D_K \cdot D_K \cdot M + M \cdot N}$$

$$= \frac{(\alpha {D_K}^2 + \alpha^2 N) \cdot \delta}{{D_K}^2 + N} \cong \alpha^2 \cdot \delta$$

(b) The ratio of computational cost:

$$\frac{D_K \cdot D_K \cdot (\alpha M \cdot \delta) \cdot D_F \cdot D_F + (\alpha M \cdot \delta) \cdot \alpha N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}$$

$$= \frac{(\alpha {D_K}^2 + \alpha^2 N) \cdot \delta}{{D_K}^2 + N} \cong \alpha^2 \cdot \delta$$

For example, suppose the N = 128, M = 64, D_K = 3, D_F = 32, α = 0.5 and δ = 2, the ratio of amount of parameters between modified depthwise separable convolution and the original one i.e. [α = 1 and δ = 1] is equal to 0.53 ≅ 0.5 and the ratio of computational cost is also equal to 0.53 ≅ 0.5.

B. *MobileNet combined with Max Pooling Kernel size 3×3 Stride 2 in different α and δ architecture*

MobileNet [1] does not use any pooling, but pooling can keep more features than only using stride in the convolution layer. So we tried to add pooling in the MobileNet [1] and expected it may raise the accuracy of recognition. Then we replaced the original stride in depthwise separable convolutions with Max Pooling Kernel size 3 × 3 Stride 2 and also adjusted the α and δ mentioned in III.**A**. The model structure is shown in **Table 2**.

Table 2. MobileNet combined with Max Pooling Kernel size 3 × 3 Stride 2 in different α and δ architecture

Type / Stride	Filter Shape	Input Size
Conv / s1	**3 × 3 × 3 × 32α**	**32 × 32 × 3**
Maxpool / s2	**Max pooling kernel 3×3 stride2**	**32 × 32 × 32α**
Conv dw / s1	**3 × 3 × (32α× δ)**	**16 × 16 × 32α**
Conv / s1	**1 × 1 × (32α× δ) × 64α**	**16 × 16 × (32α× δ)**
Maxpool / s2	**Max pooling kernel 3×3 stride2**	**16 × 16 × 64α**
Conv dw / s1	**3 × 3 × (64α× δ)**	**8 × 8 × 64α**
Conv / s1	**1 × 1 × (64α× δ) × 128α**	**8 × 8 × (64α× δ)**
Conv dw / s1	**3 × 3 × (128α× δ)**	**8 × 8 × 128α**
Conv / s1	**1 × 1 × (128α× δ) × 128α**	**8 × 8 × (128α× δ)**
Maxpool / s2	**Max pooling kernel 3×3 stride2**	**8 × 8 × 128α**
Conv dw / s1	**3 × 3 × (128α× δ)**	**4 × 4 × 128α**
Conv / s1	**1 × 1 × (128α× δ) × 256α**	**4 × 4 × (128α× δ)**
Conv dw / s1	**3 × 3 × (256α× δ)**	**4 × 4 × 256α**
Conv / s1	**1 × 1 × (256α× δ) × 256α**	**4 × 4 × (256α× δ)**
Maxpool / s2	**Max pooling kernel 3×3 stride2**	**4 × 4 × 256α**
Conv dw / s1	**3 × 3 × (256α× δ)**	**2 × 2 × 256α**
Conv / s1	**1 × 1 × (256α× δ) × 512α**	**2 × 2 × (256α× δ)**
5 ×		
Conv dw / s1	**3 × 3 × (512α× δ)**	**2 × 2 × 512α**
Conv / s1	**1 × 1 × (512α× δ) × 512α**	**2 × 2 × (512α× δ)**
Maxpool / s2	**Max pooling kernel 3×3 stride2**	**2 × 2 × 512α**
Conv dw / s1	**3 × 3 × (512α× δ)**	**1 × 1 × 512α**
Conv / s1	**1 × 1 × (512α× δ) × 1024α**	**1 × 1 × (512α× δ)**
Conv dw / s1	**3 × 3 × (1024α× δ)**	**1 × 1 × 1024α**
Conv / s1	**1 × 1 × (1024α× δ) × 1024α**	**1 × 1 × (1024α× δ)**
Avg pool / s1	**Global average pooling**	**1 × 1 × 1024α**
FC / s1	**1 × 1 × 1024α× classnumber**	**1 × 1 × 1024α**
Softmax / s1	**Classifier**	**1 × 1 × classnum**

C. MobileNet combined with Fractional Max Pooling Stride 1.4 in different α and δ architecture

Because the Fractional Max Pooling [2] can keep more feature maps than general pooling, we also tried to add Fractional Max Pooling in MobileNet [1]. We got rid of all strides in depthwise separable convolution and added Fractional Max Pooling Stride 1.4 in MobileNet [1]. Then we also adjusted the α and δ mentioned in III.**A**. The model structure is shown in **Table 3**.

IV. EXPERIMENTS

In this section, we will show the image classification results on MobileNet [1] which only adjust the α and δ, MobileNet [1] combining with Max Pooling Kernel size 3 × 3 Stride 2 in different α and δ and MobileNet [1] combining with Fractional Max Pooling Stride 1.4 in different α and δ.

All networks models were trained in Keras [13] which backend is Tensorflow [14] using the optimizer named Adam [15] with 10% dropout in a fully connection layer. We used classification database CIFAR-10 consisting of images drawn from 10 classes and CIFAR-100 from 100 classes.

The training and test sets of CIFAR-10 and CIFAR-100 respectively contain 50,000 and 10,000 images without any data augmentation.

A. MobileNet in different α and δ

We adjusted α and δ on MobileNet [1] and tested improved models respectively on CIFAR-10 and CIFAR-100. After the experiments, results show that though the computational cost and amount of parameters are increased by increasing δ, the overall computational cost and amount of parameters can be reduced by introducing α. When α = 0.5 and δ = 2, the overall parameters and computational cost can be reduced by half and the accuracy can achieve similar to the baseline MobileNet [1] which α = 1 and δ = 1. The experimental results are shown in **Table 4.**

Table 3. MobileNet combined with Fractional Max Pooling Stride 1.4 in different α and δ architecture

Type / Stride	Filter Shape	Input Size
Conv / s1	**3 × 3 × 3 × 32α**	**32 × 32 × 3**
FMP / s1.4	**Fractional Max Pooling s1.4**	**32 × 32 × 32α**
Conv dw / s1	**3 × 3 × (32α× δ)**	**22 × 22 × 32α**
Conv / s1	**1 × 1 × (32α× δ) × 64α**	**22 × 22 × (32α× δ)**
Conv dw / s1	**3 × 3 × (64α× δ)**	**22 × 22 × 64α**
Conv / s1	**1 × 1 × (64α× δ) × 128α**	**22 × 22 × (64α× δ)**
FMP / s1.4	**Fractional Max Pooling s1.4**	**22 × 22 × 128α**
Conv dw / s1	**3 × 3 × (128α× δ)**	**15 × 15 × 128α**
Conv / s1	**1 × 1 × (128α× δ) × 128α**	**15 × 15 × (128α× δ)**
Conv dw / s1	**3 × 3 × (128α× δ)**	**15 × 15 × 128α**
Conv / s1	**1 × 1 × (128α× δ) × 256α**	**15 × 15 × (128α× δ)**
FMP / s1.4	**Fractional Max Pooling s1.4**	**15 × 15 × 256α**
Conv dw / s1	**3 × 3 × (256α× δ)**	**10 × 10 × 256α**
Conv / s1	**1 × 1 × (256α× δ) × 256α**	**10 × 10 × (128α× δ)**
Conv dw / s1	**3 × 3 × (256α× δ)**	**10 × 10 × 256α**
Conv / s1	**1 × 1 × (256α× δ) × 512α**	**10 × 10 × (256α× δ)**
FMP / s1.4	**Fractional Max Pooling s1.4**	**10 × 10 × 512α**
2 ×		
Conv dw / s1	**3 × 3 × (512α× δ)**	**6 × 6 × 512α**
Conv / s1	**1 × 1 × (512α× δ) × 512α**	**6 × 6 × (512α× δ)**
FMP / s1.4	**Fractional Max Pooling s1.4**	**6 × 6 × 512α**
2 ×		
Conv dw / s1	**3 × 3 × (512α× δ)**	**4 × 4 × 512α**
Conv / s1	**1 × 1 × (512α× δ) × 512α**	**4 × 4 × (512α× δ)**
FMP / s1.4	**Fractional Max Pooling s1.4**	**4 × 4 × 512α**
Conv dw / s1	**3 × 3 × (512α× δ)**	**2 × 2 × 512α**
Conv / s1	**1 × 1 × (512α× δ) × 512α**	**2 × 2 × (512α× δ)**
Conv dw / s1	**3 × 3 × (512α× δ)**	**2 × 2 × 512α**
Conv / s1	**1 × 1 × (512α× δ) × 1024α**	**2 × 2 × (512α× δ)**
FMP / s1.4	**Fractional Max Pooling s1.4**	**2 × 2 × 1024α**
Conv dw / s1	**3 × 3 × (1024α× δ)**	**1 × 1 × 1024α**
Conv / s1	**1 × 1 × (1024α× δ) × 1024α**	**1 × 1 × (1024α× δ)**
Avg Pool / s1	**Global average pooling**	**1 × 1 × 1024α**
FC / s1	**1 × 1 × 1024α× classnumber**	**1 × 1 × 1024α**
Softmax / s1	**Classifier**	**1 × 1 × classnum**

Table 4. Results on MobileNet in different α and δ

Model MobileNet	CIFAR-10 Accuracy	Million Mult-Adds	Million Parameters
δ = 1,α= 1 Baseline	**76.7%**	**11.58**	**3.23**
δ = 2,α= 1	**81.1%**	**22.96**	**6.44**
δ = 2,α= 0.5	**74.9%**	**5.97**	**1.65**
δ = 2,α= 0.25	**70.2%**	**1.61**	**0.43**
δ = 4,α= 1	**81.7%**	**45.69**	**12.85**
δ = 4,α= 0.5	**78.6%**	**11.83**	**3.28**
δ = 4,α= 0.25	**73.8%**	**3.16**	**0.85**
Model MobileNet	CIFAR-100 Accuracy	Million Mult-Adds	Million Parameters
δ = 1,α= 1 Baseline	**39.1%**	**11.68**	**3.33**
δ = 2,α= 1	**41.8%**	**22.97**	**6.53**
δ = 2,α= 0.5	**38.2%**	**5.98**	**1.70**
δ = 2,α= 0.25	**37.1%**	**1.62**	**0.46**
δ = 4,α= 1	**44.3%**	**45.70**	**12.94**
δ = 4,α= 0.5	**43.5%**	**11.84**	**3.33**
δ = 4,α= 0.25	**37.9%**	**3.17**	**0.88**

Table 5. Results on MobileNet combined with Max Pooling Kernel size 3 × 3 Stride 2 in different α and δ

Model MobileNet+MP3	CIFAR-10 Accuracy	Million Mult-Adds	Million Parameters
δ = 2,α= 1	**81.3%**	**23.18**	**6.44**
δ = 2,α= 0.5	**77.3%**	**6.08**	**1.65**
δ = 2,α= 0.25	**71.2%**	**1.67**	**0.43**
δ = 4,α= 1	**82.3%**	**45.91**	**12.85**
δ = 4,α= 0.5	**79.7%**	**11.94**	**3.28**
δ = 4,α= 0.25	**74.8%**	**3.22**	**0.85**
Baseline MobileNet	**76.7%**	**11.58**	**3.23**
Model MobileNet+MP3	CIFAR-100 Accuracy	Million Mult-Adds	Million Parameters
δ = 2,α= 1	**42.3%**	**23.28**	**6.53**
δ = 2,α= 0.5	**41.0%**	**6.13**	**1.70**
δ = 2,α= 0.25	**38.1%**	**1.64**	**0.46**
δ = 4,α= 1	**46.2%**	**46.01**	**12.94**
δ = 4,α= 0.5	**43.5%**	**12.0**	**3.33**
δ = 4,α= 0.25	**37.9%**	**3.24**	**0.88**
Baseline MobileNet	**39.1%**	**11.68**	**3.33**

Table 6. Results on MobileNet combined with Fractional Max Pooling Stride 1.4 in different α and δ

Model MobileNet+FMP	CIFAR-10 Accuracy	Million Mult-Adds	Million Parameters
δ = 2,α= 1	**88.9%**	**139**	**6.44**
δ = 2,α= 0.5	**86.1%**	**35.9**	**1.65**
δ = 2,α= 0.25	**82.1%**	**9.5**	**0.43**
δ = 4,α= 1	**89.6%**	**277**	**12.85**
δ = 4,α= 0.5	**87.8%**	**71.2**	**3.28**
δ = 4,α= 0.25	**84.5%**	**18.9**	**0.85**
Baseline MobileNet	**76.7%**	**11.58**	**3.23**
Model MobileNet+FMP	CIFAR-100 Accuracy	Million Mult-Adds	Million Parameters
δ = 2,α= 1	**59.3%**	**139**	**6.53**
δ = 2,α= 0.5	**56.9%**	**36**	**1.70**
δ = 2,α= 0.25	**50.8%**	**9.6**	**0.46**
δ = 4,α= 1	**60.9%**	**277**	**12.94**
δ = 4,α= 0.5	**58.6%**	**71.3**	**3.33**
δ = 4,α= 0.25	**54.1%**	**18.9**	**0.88**
Baseline MobileNet	**39.1%**	**11.68**	**3.33**

B. MobileNet combined with Max Pooling Kernel size 3×3 Stride 2 in different α and δ

We combined Max Pooling Kernel size 3 × 3 Stride 2 with MobileNet [1] and tested modified models on CIFAR- 10 and CIFAR-100. After the experiments, We found that the accuracy raised with increasing few computational cost. But if we adjusted the appropriate α and δ, e.g. [α = 0.5 and δ = 2] at the same time, we can achieve higher accuracy than the baseline MobileNet [1] which α = 1 and δ = 1 as well as using less computational cost and parameters. The experimental results are shown in **Table 5.**

C. MobileNet combined with Fractional Max Pooling Stride 1.4 in different α and δ

We integrated Fractional Max Pooling Stride 1.4 with MobileNet [1] and tested modified models on CIFAR-10 and CIFAR-100. We found that the accuracy is raised with increasing the huge amount of computational cost. But if we adjusted the appropriate α and δ, e.g. [α = 0.25 and δ = 2] at the same time, we can obtain much higher accuracy than the baseline MobileNet [1] which α = 1 and δ = 1 as well as using less computational cost and parameters. The experimental results are shown in **Table 6.**

Table 7. Comparison of the better experimental results

Model	CIFAR-10 Accuracy	Million Mult-Adds	Million Parameters
MobileNet (Baseline) δ = 1,α= 1	**76.7%**	**11.58**	**3.23**
MobileNet δ = 2,α= 0.5	**74.9%**	**5.97**	**1.65**
MobileNet+MP3 δ = 2,α= 0.5	**77.3%**	**6.08**	**1.65**
MobileNet+FMP δ = 2,α= 0.25	**82.1%**	**4.8**	**0.22**
Model	CIFAR-100 Accuracy	Million Mult-Adds	Million Parameters
MobileNet (Baseline) δ = 1,α= 1	**39.1%**	**11.68**	**3.33**
MobileNet δ = 2,α= 0.5	**38.2%**	**5.98**	**1.70**
MobileNet+MP3 δ = 2,α= 0.5	**41.0%**	**6.13**	**1.70**
MobileNet+FMP δ = 2,α= 0.25	**50.8%**	**9.6**	**0.46**

D. Comprehensive comparison table

We chose the better results on above experiments, e.g. [MobileNet in α = 0.5 and δ = 2, MobileNet+MP3 in α = 0.5 and δ = 2 and MobileNet+FMP in α = 0.25 and δ = 2] to made comparison which baseline is MobileNet [1] in α = 1 and δ =1. The comparison results are shown in **Table 7.**

V. Conclusion

In this paper, we proposed a new model architecture to improve the MobileNet [1]. The new networks model is based on adjusting both the δ called depth multiplier and the α called width multiplier in the MobileNet [1] and integrating either the Max Pooling Kernel size 3 × 3 Stride 2 or the Fractional Max Pooling [2] Stride 1.4 with the MobileNet [1]. Compared with the MobileNet [1], the proposed architecture not only result in a higher accuracy but also reducing the computational cost and the huge amount of parameters. We recommend to use MobileNet [1] combining with the Fractional Max Pooling [2] Stride 1.4 in α = 0.25 and δ = 2 or MobileNet [1] combining with Max Pooling Kernel size 3 × 3 Stride 2 in α = 0.5 and δ = 2 for the device is not support for Fractional Max Pooling [2].

REFERENCES

[1] A. G. Howard, M. Zhu,. B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam. MobileNet : Efficient Convolutional Neural Networks for Mobile Applications. *arXiv preprint arXiv:1704.04861,2017.*

[2] B. Graham.Fractional Max-Pooling. *arXiv preprint arXiv: 1412.6071, 2014.*

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet Classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge International Journal of Computer Vision, 115(3):211-252, 2015.

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

[6] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *arXiv preprint arXiv:1512.00567,2015*.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv :1512.03385,2015.*

[8] G. Huang, Z. Liu, K. Q. Weinberger and L. Maaten. Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993,2016.*

[9] J. Jin, A. Dundar,.and E. Culurciello. Flattened convolutional neural networks for feedforward acceleration.*arXiv preprint arXiv: 1412. 5474, 2014.*

[10] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally and K. Keutzer. SqueezeNet : AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size. *arXiv preprint arXiv:1602. 07360,2016.*

[11] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnornet : Imagenet classification using binary convolutional neural networks. a*rXiv preprint arXiv:1603.05279,2016.*

[12] M. Wang, B. Liu, and H. Foroosh. Factorized convolutional neural networks. *arXiv preprint arXiv: 1608.04337, 2016.*

[13] @misc{chollet2015keras, title={Keras}, author={Chollet, ran\c{c}ois and others}, year={2015}, publisher={GitHub}, howpublished ={\ur l{https://github.com/fchollet/keras}}, }.

[14] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M.Devin, et al. Tensorflow:Large-scale machine learning onheterogeneous systems, 2015. Software available from tensorflow org, 1,2015.

[15] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv: 1412.6980,2014.*