Practical Analysis on Architecture of EfficientNet

Van-Thanh Hoang
Faculty of Engineering-Technology
Quang Binh University
Quang Binh, Vietnam
Email: thanhhv@qbu.edu.vn
ORCID: 0000-0003-3478-9954

Kang-Hyun Jo
School of Electrical Engineering
University of Ulsan
Ulsan, Korea
Email: acejo@ulsan.ac.kr
ORCID: 0000-0001-8317-6092

***Abstract*—Convolutional neural networks (CNNs) have shown significant performance in solving various artificial intelligence tasks in recent years. However, the increasing model size has raised challenges in adopting them in limited-resource applications. Recently, many research works try to build efficient networks which are as small as possible while still have acceptable performance. The state-of-the-art architecture is EfficientNet. On the ImageNet challenge, with a much fewer parameter calculation load, EfficientNet could take its place among the state-of-the-art. EfficientNet can be considered a group of convolutional neural network models. But given some of its subtleties, it's actually more efficient than most of its predecessors. It uses the inverted bottleneck residual blocks of MobileNetV2, in addition to squeeze-and-excitation modules (SE modules). This paper investigate the effect of SE modules on the performance of EfficientNet-B0, the basic network model in the EfficientNets series, by repositioning/removing the SE modules.**

***Index Terms*—CNN architecture, efficientnet, SE module**

I. Introduction

Deep convolutional neural networks (CNNs) have shown significant performance in many computer vision tasks in recent years. The primary trend for solving major tasks is building deeper and larger CNNs [2], [12]. The most accurate CNNs usually have hundreds of layers and thousands of channels [2], [5], [13], [16]. Many real-world applications need to be performed in real-time and/or on limited-resource mobile devices. Thereby, the model should be compact and low computational cost. The model compression work is actually investigating the trade-off between efficiency and accuracy.

Recently, many research works focus on the field of designing efficient architecture. Inspired by the architecture proposed in [7], the Inception module is proposed in GoogLeNet [12] to build deeper networks without increase model size and computational cost. Then it is further improved in [13] through factorizing convolution. The Depthwise Separable Convolution (DWConvolution) generalized the factorization idea and decomposed the standard Convolution into a depthwise convolution followed by a pointwise 1×1 convolution. EfficientNets [15], a family of efficiently scaled convolutional neural nets, have recently emerged as state-of-the-art models for image classification tasks. EfficientNets optimize for accuracy as well as efficiency by reducing model size and floating point operations executed while still maintaining model quality. The basic network architecture of the model is designed by using neural architecture search. By enlarging the basic model of EfficientNets, a series of EfficientNets models are obtained. This series of models defeated all previous convolutional neural network models in terms of efficiency and accuracy. In particular, EfficientNet-B7 obtained top-1 accuracy of 84.4% and top-5 accuracy of 97.1% on the ImageNet dataset. And compared with other models with the highest accuracy at the time, the size was reduced by 8.4 times and the efficiency was increased by 6.1 times. And through transfer learning, EfficientNets reached the most advanced level at the time on multiple well-known data sets.

This paper investigates the EfficientNet-B0, the basic network model in the EfficientNets series. The core structure of the network is the mobile inverted bottleneck convolution (MBConv) module, which also introduces the attention thought of the compression and excitation network (Squeeze-and-Excitation Network, SENet) [4]. We create some variations of EfficientNet-B0 by repositioning/removing SE modules. All variants are then trained on ImageNet dataset [10] to know the effect of SE modules on the performance of EfficientNet-B0.

II. Related Work and Background

A. Related Work

Recently, there are many studies focus on the designing efficient architectures approach [3], [5], [11], [17], [18]. They have explored efficient CNNs that can be trained end-to-end.

MobileNetV1 [3] introduced *depthwise separable convolutions* as an efficient replacement for traditional convolution layers. Depthwise separable convolutions effectively factorize traditional convolution by separating spatial filtering from the feature generation mechanism. It consists of two separated layers. The first layer uses a light weight depthwise convolution operator. It applies a single convolutional filter per input channel to capture the spatial information in each channel. Then the second layer employs a pointwise convolution, means a heavier 1×1 convolution, to capture the cross-channel information for feature generation.

MobileNetV2 [11] introduced the linear bottleneck and inverted residual structure in order to make even more efficient layer structures by leveraging the low rank nature of the problem. This structure is shown on Figure 1a and is defined by a 1×1 expansion convolution followed by depthwise

convolutions and a 1×1 projection layer. The input and output are connected with a residual connection if and only if they have the same number of channels and spatial sizes. This structure maintains a compact representation at the input and the output while expanding to a higher-dimensional feature space internally to increase the expressiveness of nonlinear perchannel transformations.

EfficientNets [15], a family of efficiently scaled convolutional neural nets, have recently emerged as state-of-the-art models for image classification tasks. EfficientNets optimize for accuracy as well as efficiency by reducing model size and floating point operations executed while still maintaining model quality.

The architecture of EfficientNet was designed by performing the neural architecture search, a technique for automating the design of neural networks. It optimizes both the accuracy and efficiency as measured on the floating-point operations per second (FLOPS) basis. This developed architecture uses the mobile inverted bottleneck convolution (MBConv). The researchers then scaled up this baseline network to obtain a family of deep learning models, called EfficientNets. The architecture of EfficientNet-B0, the basic network model in the EfficientNets series, is given in the Table I.

The efficiency is the utmost important problem for mobile-size convolution model. The efficient operations have been extensively studied in the MobileNet family [11], [14]. Spare depthwise convolution and the inverted bottleneck block are the core ideas for efficient mobile size network. MnasNet [14] and EfficientNet [15] takes a step further to develop MBConv operation based on the mobile inverted bottleneck in [11]. EfficientNet shows that the models with MBConv operations not only achieving the state-of-the-art in ImageNet challenge but also very efficient.

B. Squeeze-and-Excitation Block

Squeeze-and-Excitation Networks [4] introduce a building module for CNNs that improves channel interdependencies at almost no computational cost. CNNs use their convolutional filters to extract hierarchal information from images. Lower layers find trivial pieces of context like edges or high frequencies, while upper layers can detect faces, text or other complex geometrical shapes. They extract whatever is necessary to solve a task efficiently.

All of this works by fusing the spatial and channel information of an image. The different filters will first find spatial features in each input channel before adding the information across all available output channels.

All you need to understand for now is that the network weights each of its channels equally when creating the output feature maps. SENets are all about changing this by adding a content aware mechanism to weight each channel adaptively. In it's most basic form this could mean adding a single parameter to each channel and giving it a linear scalar how relevant each one is.

However, the authors push it a little further. First, they get a global understanding of each channel by squeezing the feature maps to a single numeric value. This results in a vector of size n, where n is equal to the number of convolutional channels. Afterwards, it is fed through a two-layer neural network, which outputs a vector of the same size. These n values can now be used as weights on the original features maps, scaling each channel based on its importance.

Specifictly, at first, they squeeze each channel to a single numeric value using average pooling. Then, a fully connected layer followed by a ReLU function adds the necessary non-linearity. It's output channel complexity is also reduced by a certain ratio. After that, a second fully connected layer followed by a Sigmoid activation gives each channel a smooth gating function. At last, they weight each feature map of the convolutional block based on the result of the side network. These four steps add almost no additional computing cost (less than 1%) and can be added to any model.

The authors show that by adding SE modules to ResNet-50 you can expect almost the same accuracy as ResNet-101 delivers. This is impressive for a model requiring only half of the computational costs. The paper further investigates other architectures like Inception, Inception-ResNet and ResNeXt. The latter leads them to a modified version that shows a top-5 error of 3.79% on ImageNet.

TABLE I
EFFICIENTNET-B0 BODY ARCHITECTURE. **MBCONV**: MOBILE INVERTED BOTTLENECK CONVOLUTION, THE FOLLOWING NUMBER IS THE EXPANDED FACTOR.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Blocks $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling	7×7	1280	1
10	FC	1×1	1000	1

III. NETWORK ARCHITECTURE

A. Architecture of EfficientNet-B0

EfficientNet-B0 is the basic network model in the EfficientNets series. The core structure of the network is the mobile inverted bottleneck convolution (MBConv) module, which also introduces the attention thought of the compression and excitation module [4]. Its architecture is shown in Table I. All layers are followed by a batchnorm and ReLU nonlinearity activation with the exceptionof the final fully connected layer which has no nonlinearity and feeds into a softmax layer for classification. Down sampling is handled with strided convolution in the depthwise convolutions of first block in each stage, as well as in the first layer. A final average pooling reduces the spatial resolution to 1 before the fully connected layer.

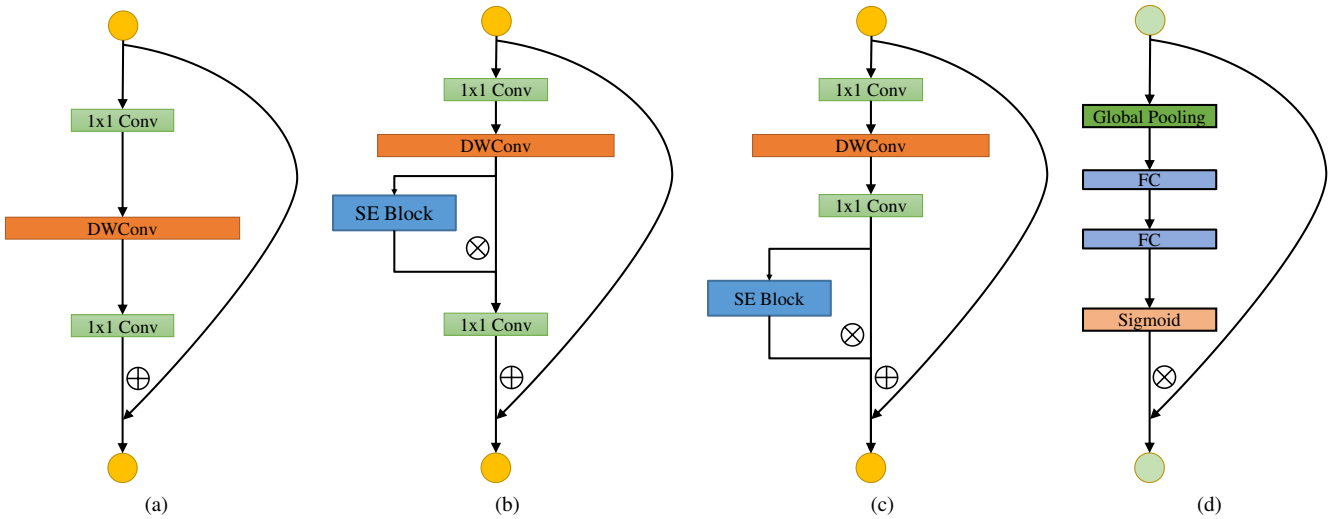

Fig. 1. Architecture of building block. a) The inverted residual block of MobileNetV2 [11] (which without SE module); b) The inverted residual block with SE module behind the main Depthwise Convolution of EfficientNet [15]. c) The inverted residual block with SE module after the 1×1 Conv. d) The SE module. The down sampling will be in the DWConv layer. There is no shortcut connection if the sizes of input and output features are different. **Conv**: Convolution layer. **DWCon**: Depthwise Convolution layer.

B. Mobile Inverted Bottleneck Convolution Block

The basic mobile inverted bottleneck convolution is obtained through the neural network architecture search. The module structure is similar to the depthwise separable convolution, as can be seen in Fig. 1a. The mobile inverted bottleneck convolution first performs a 1×1 point-by-point convolution on the input and based on the expansion ratio (expand ratio) to change the output channel dimension (for example, when the expansion ratio is 3, the channel dimension will be increased by 3 times. But if the expansion ratio is 1, then directly omit the 1×1 point-by-point convolution and subsequent batch normalization and activation functions). Then proceed to the depthwise convolution of $k \times k$. Then restore the original channel dimension at the end of 1x1 point-by-point convolution. Finally, the drop connect and skip connection of the input are performed.

When the SE module is attached, the operations will be performed after the depthwise convolution $k \times k$, as shown in Fig. 1b. This architecture is default for EfficientNet-B0, as well as the EfficientNet series.

This paper also investigates the performance of EfficientNet-B0 when the SE module is moved behind the last 1×1 convolution layer. The architecture of this block is shown in Fig. 1c.

IV. Experiments

This paper evaluates the variants of EfficientNet-B0 on ImageNet dataset [10] and compare them in term of Top-1 error.

A. Dataset

The ILSVRC 2012 classification dataset [10] consists 1.2 million images for training, and $50,000$ for validation, from $1,000$ classes. This paper adopts the same data augmentation scheme for training images as in [15], and apply a single-crop with size 224×224 at test time. Following [15], we report the Top-1 classification errors on the validation set.

B. Implementation Details

This paper implements all variants of EfficientNet-B0 on Tensorflow. The training procedure follows the schema proposed in [15]. All models are trained using back-propagation [6] by Stochastic Gradient Descent [9] with Nesterov momentum [8] (NAG) optimizer implemented by Tensorflow for 350 epochs. At the first 5 epochs, the learning rate increases linearly from 0 to 0.4. Then, its value will be decayed with a cosine shape (the learning rate of epoch $t \leq 300$ is set to $0.5 \times \text{lr} \times (\cos(\pi \times t/300) + 1)$). The parameters are initialized by Xavier's initializer [1]. The other settings are: weight decay of 0.0001, momentum of 0.9, and batch size of 2048.

All networks were trained on Google Colab service with TPU environment.

C. Performance Evaluation

Table II shows the comparison on Top-1 error of variants of EfficientNet-B0 when the SE modules are repositioned/removed.

It's easy to see that if we do the change (re-positioning or removing) on noskip blocks, the accuracy slightly drops with just 0.1%. But if we do the same change on hasskip blocks, the accuracy drops a little bit larger at 0.7%. If all blocks are applied, the accuracy drops much at 1% and 1.4%.

One note here is if we reposition the SE modules on only noskip-blocks or hasskip-blocks, the performance is similar to the removing case. It means the affect of SE modules when it's at the end almost is absent. However, when the change is applied to all blocks, the variant with SE blocks has higher accuracy, 76.2% vs 75.8%. This shows that SE modules still

TABLE II

TOP-1 ERROR RATES (%) VARIANTS OF EFFICIENTNET ON IMAGENET DATASETS. RESULTS THAT OUTPERFORM ARE **BOLD**. *SEend* MEANS SE MODULES ARE REPOSITIONED TO THE END OF MBCONV BLOCKS. *noSE* MEANS THERE IS NO SE MODULES INSIDE THE MBCONV BLOCKS. *noskip* MEANS THE CHANGES ARE APPLIED TO MBCONV BLOCKS WHICH HAVE NO SKIP-CONNECTION. *hasskip* MEANS THE CHANGES ARE APPLIED TO MBCONV BLOCKS WHICH HAVE SKIP-CONNECTION.

Model	Block Arch.	#Params	Top-1	Top-5
EfficienNet-B0	Fig. 1b	5.29M	77.2	93.4
EfficienNet-B0-SEend@*noskip*	Fig. 1c	5.16M	77.1	93.3
EfficienNet-B0-SEend@*hasskip*		4.91M	76.5	93.2
EfficienNet-B0-SEend@*all*		4.78M	76.2	92.9
EfficienNet-B0-noSE@*noskip*	Fig. 1a	5.11M	77.1	93.4
EfficienNet-B0-noSE@*hasskip*		4.83M	76.6	93.0
EfficienNet-B0-noSE@*all*		4.65M	75.8	92.7

can improve the accuracy for EfficientNet-B0, but we don't need to use SE module for all MBConv blocks.

V. CONCLUSION

This paper investigate the effect of SE modules on the performance of EfficientNet-B0, the basic network model in the EfficientNets series, by repositioning/removing the SE modules. The experiments show that repositioning SE module to the end of block is not a good idea. And we don't need to use SE module for all MBConv blocks, there is a trade off need to be further studied to have more efficient network.

In the future, it is necessary to modify the architecture of EfficientNet and MBConv block to make them more efficient.

REFERENCES

[1] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[3] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017, arXiv preprint arXiv:1704.04861.

[4] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.

[5] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[6] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[7] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proceedings of the International Conference on Learning Representations*, 2014.

[8] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate o (1/kˆ 2)," in *Dokl. Akad. Nauk SSSR*, vol. 269, 1983, pp. 543–547.

[9] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[11] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[13] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[14] M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le, "Mnasnet: Platform-aware neural architecture search for mobile," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2820–2828.

[15] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the International Conference on Machine Learning*, 2019, pp. 6105–6114.

[16] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proceedings of the British Machine Vision Conference*, 2016.

[17] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6848–6856.

[18] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8697–8710.