EFFICIENTNET-ENHANCED AVIAN SPECIES IDENTIFICATION AND MONITORING SYSTEM

Yamuna S[1], Gayathri K[2], Srimathi T[3], Anbu Thuran[4].Jayavishnusivam R[5]

[1] *Assistant Professor, Department of Information Technology, Bannari Amman Institute of Technology, Tamil Nadu, India*
[2] *Assistant Professor, Department of Information Technology, Bannari Amman Institute of Technology, Tamil Nadu, India*
[3] *Student, Department of Information Technology, Bannari Amman Institute of Technology, Tamil Nadu, India*
[4] *Student, Department of Computer Science and Engineering, Bannari Amman Institute of Technology, Tamil Nadu, India*
[5] *Student, Department of Information Technology, Bannari Amman Institute of Technology, Tamil Nadu, India*

ABSTRACT

The birds have been with us for many years. There are many disasters happening at every moment that lead to the disturb survival of birds. Considering as a friend it is our duty to save them and make their life possible on the Earth. Convolutional Neural Networks (ConvNets) are typically developed with a fixed resource budget and then scaled up for improved accuracy if more resources are available. In this paper, we investigate model scaling systematically and discover that carefully balancing network depth, width, and resolution can lead to improved performance. Based on this observation, we propose a new scaling method that uses a simple yet highly effective compound coefficient to uniformly scale all depth/width/resolution dimensions. We show how effective this method is for scaling up MobileNets and ResNet.

Keyword : - *Birds, EfficientNet, CNN, Identification, Monitor*

1. Introduction

Keras efficient net is a fast model that achieves state-of-the-art accuracy in common image classification and imagenet for transfer learning tasks. The keras smaller efficientnet model is comparable to the Mnasnet, which was similar to the SOTA, which was a smaller model. It offers multiple models, beginning with B0 and ending with B7, which represent the combination of accuracy and efficiency on a wide range of scales. The keras efficientnet function returns the keras image classification model that has been pre trained with imagenet and optionally loaded with weights. Every keras application expects a specific type of input preprocessing, and for efficientnet preprocessing, input is included. As a result, the efficientnet preprocessing input is being processed by the function.

The model expects their inputs to be float tensor pixels with values ranging from 0 to 255. This function returns the Keras model, which may or may not have been loaded with weights. The model will optimize the efficiency and accuracy of floating point operation. According to stat from Google AI Blog Training efficiency has gained significant interests recently. Our EfficientNetV2 performs well on ImageNet, CIFAR-10, CIFAR100, Cars, and Flowers datasets thanks to improved progressive learning. We achieve 85.7% top-1 accuracy on ImageNet while training 3x - 9x faster and being up to 6.8x smaller than previous models (Figure 1.1). Our EfficientNetV2 and progressive learning also make training models on larger datasets easier. ImageNet21k (Russakovsky et al., 2015) is approximately 10 times larger than ImageNet ILSVRC2012, but our EfficientNetV2 can complete the training in two days using moderate computing resources of 32 TPUv3 cores. Our EfficientNetV2 achieves 87.3% top-1

accuracy on ImageNet ILSVRC2012 after pretraining on the public ImageNet21k, outperforming the recent ViT-L/16 by 2.0% accuracy while training 5x-11x faster (Figure 1.1). We make three contributions: 1. We introduce EfficientNetV2, a new family of smaller and faster models. EfficientNetV2 outperforms previous models in terms of training speed and parameter efficiency, as discovered by our training-aware NAS and scaling. 2. We suggest a more effective progressive learning technique that adaptively modifies both regularization and image size. We demonstrate that it expedites training while also increasing accuracy. 3. On ImageNet, CIFAR, Cars, and Flowers datasets, we show up to 11x quicker training speed and up to 6.8x higher parameter efficiency than prior art.

Architecture	Performance
EfficientNetB0	0.788
EfficientNetB1	0.815
EfficientNetB2	0.824
EfficientNetB3	0.841
EfficientNetB4	0.853
EfficientNetB5	0.861
EfficientNetB6	0.864

Table 1.1

2. . Related work

Training and parameter efficiency: Numerous research, such DenseNet (Huang et al., 2017) and EfficientNet (Tan & Le, 2019a), put a strong emphasis on parameter efficiency with the goal of improving accuracy while using fewer parameters. Instead of focusing on parameter efficiency, several more recent research try to increase training or inference speed. For instance, RegNet (Radosavovic et al., 2020), ResNeSt (Zhang et al., 2020), TResNet (Ridnik et al., 2020), and EfficientNet-X (Li et al., 2021) concentrate on increasing the speed of GPU and/or TPU inference, whereas NFNets (Brock et al., 2021) and BoTNets (Srin Their training or inference speed, though, frequently comes at the expense of more parameters. With respect to the state of the art, this work attempts to greatly increase training speed and parameter efficiency.

Progressive training: Various types of progressive training, which dynamically change the training settings or networks, have been proposed in previous works for GANs, transfer learning, adversarial learning, and language models (Press et al., 2021). The term "progressive resizing" (Howard, 2018) primarily refers to our strategy, which strives to increase training efficiency. However, it frequently comes at a cost to accuracy. Mix&Match (Hoffer et al., 2019), which randomly samples different image sizes for each batch, is another closely similar piece of work. Both progressive resizing and Mix&Match reduce accuracy by utilizing the same regularization for all image sizes.

3. Scaling Dimensions

The primary challenge is that the optimal d, w, and r depend on one another and change in value under various resource limitations. Because of this challenge, conventional ConvNets are typically scaled in one of these dimensions:

3.1 Depth(d): The most prevalent method employed by many ConvNets is scaling the network depth. It seems to make sense that a deeper ConvNet can generalize effectively to new tasks and collect richer, more complicated characteristics. However, because of the vanishing gradient issue, deeper networks are likewise more challenging to train. Skip connections and batch normalization are two methods that help with the training problem, but the

accuracy gain of very deep networks declines as a result. For instance, ResNet-1000 has similar accuracy to ResNet-101 while having many more layers. Our empirical analysis on scaling a baseline model with various depth coefficients is shown in Figure 3 (middle), which further illustrates the falling accuracy return for very deep ConvNets.

3.2 Width(w): For small size models, network width scaling is frequently utilized. Wider networks may typically capture more fine-grained information and are simpler to train, as was previously discussed. Extremely wide but shallow networks, on the other hand, frequently struggle to capture higher level elements. Our empirical findings in Figure 1.1 demonstrate that when networks get significantly broader with increasing w, accuracy quickly reaches saturation.

3.3 Resolution (r): ConvNets may be able to detect more minute patterns with higher resolution input images. Modern ConvNets typically use 299x299 or 331x331 starting from 224x224 in early ConvNets for higher accuracy. With 480x480 resolution, GPipe recently achieved state-of-the-art ImageNet accuracy. The accuracy gain decreases for very high resolutions (r = 1.0 denotes resolution 224x224 and r = 2.5 denotes resolution 560x560), yet higher resolutions like 600x600 are also frequently employed in object detection ConvNets. Higher resolutions do, in fact, enhance accuracy.

3.4 Compound Scaling: Different scaling dimensions are not independent, as we experimentally find. It makes sense, then, to enhance network depth for higher resolution photos so that the larger receptive fields can aid in capturing similar features that include more pixels in larger images. Consequently, when resolution is higher, we should also widen the network.

A model with a variable width coefficient is shown by each dot on a line (w). Table 1.1 contains all baseline networks. While the final baseline network (d=2.0, r=1.3) has 36 layers with a resolution of 299x299, the first baseline network (d=1.0, r=1.0) has 18 convolutional layers with a resolution of 224x224. It is essential to balance network width, depth, and resolution across all dimensions during ConvNet scaling in order to achieve higher accuracy and efficiency. In truth, some earlier research has attempted to arbitrarily balance network breadth and depth, but each of these approaches requires time-consuming human tuning. We suggest a new compound scaling technique that, according to a set of guiding principles, equally scales network breadth, depth, and resolution using a compound coefficient:

$$\begin{aligned} \text{depth: } & d = \alpha^\phi \\ \text{width: } & w = \beta^\phi \\ \text{resolution: } & r = \gamma^\phi \\ \text{s.t. } & \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\ & \alpha \geq 1, \beta \geq 1, \gamma \geq 1 \end{aligned}$$

where the results of a tiny grid search can be used to calculate the constants,, and. ,, and indicate how to allocate these additional resources to network breadth, depth, and resolution, respectively. Intuitively, is a user-specified coefficient that governs how many more resources are available for model scaling. Notably, the FLOPS of a standard convolution operation is proportional to d, w*w, and r*r, meaning that although network depth will double FLOPS, network width or resolution will cause FLOPS to grow by four times. Scaling a ConvNet with equation 3 will roughly increase total FLOPS by since convolution operations often account for the majority of the computation cost in ConvNets.

4. Results of Transfer Learning for EfficientNet

A list of frequently used transfer learning datasets was utilized to assess our EfficientNet. We use the same training parameters as and, which fine-tune on fresh datasets using ImageNet's pretrained checkpoints. The effectiveness of transfer learning is shown in Table 5: (1) With an average parameter decrease of 4.7x (up to 21x), our EfficientNet models outperform publicly accessible models like NASNet-A (Zoph et al., 2018) and Inception-v4. (2) Despite

employing 9.6x fewer parameters, our EfficientNet models still outperform state-of-the-art models like DAT (Ngiam et al., 2018), which dynamically synthesizes training data, and GPipe, which is trained utilizing specialized pipeline parallelism.

5. Discussion

The ImageNet performance of several scaling methods for the same EfficientNet-B0 baseline network in order to separate the impact of our suggested scaling approach from the EfficientNet architecture. In general, all scaling techniques increase accuracy as the cost increases. 25% more FLOPS, but our suggested compound scaling method can also increase accuracy over current single-dimension scaling methods by up to 2.5%, highlighting the significance of our compound scaling proposal. Comparing the class activation map for some example models with different scaling strategies to help clarify why our compound scaling method is superior to others. Images from the ImageNet validation set are chosen at random and scaled across all of these models from the same baseline. In contrast to previous models, which either lack object details or are unable to capture all of the objects in the photos, the model with compound scaling has a tendency to concentrate on more pertinent regions with more object features.

6. Conclusion

In this paper, we rigorously investigate ConvNet scaling and show that a crucial but absent component, precisely balancing network width, depth, and resolution, is impeding us from more accuracy and efficiency. We offer a straightforward and incredibly efficient compound scaling strategy to solve this problem, allowing us to rapidly scale up a baseline ConvNet to any target resource limitations in a more principled manner while keeping model effectiveness. We show, using both ImageNet and five widely used transfer learning datasets, that a mobilesize EfficientNet model can be scaled up very successfully, surpassing state-of-the-art accuracy with a factor of ten less parameters and FLOPS.

7. REFERENCES

[1]. Berg, T., Liu, J., Woo Lee, S., Alexander, M. L., Jacobs, D. W., and Belhumeur, P. N. Birdsnap: Large-scale fine-grained visual categorization of birds. CVPR, pp. 2011–2018, 2014.
[2]. Elfwing, S., Uchibe, E., and Doya, K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. Neural Networks, 107:3–11, 2018
[3]. Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415, 2016.
[4]. Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5 mb model size. arXiv preprint arXiv:1602.07360, 2016.
[5]. Liu, C., Zoph, B., Shlens, J., Hua, W., Li, L.-J., Fei-Fei, L., Yuille, A., Huang, J., and Murphy, K. Progressive neural architecture search. ECCV, 2018.
[6]. Nilsback, M.-E. and Zisserman, A. Automated flower classification over a large number of classes. ICVGIP, pp. 722–729, 2008.
[7]. Ramachandran, P., Zoph, B., and Le, Q. V. Searching for activation functions. arXiv preprint arXiv:1710.05941, 2018.
[8]. Real, E., Aggarwal, A., Huang, Y., and Le, Q. V. Regularized evolution for image classifier architecture search. AAAI, 2019.
[9]. Srinivas, A., Lin, T.-Y., Parmar, N., Shlens, J., Abbeel, P., and Vaswani, A. Bottleneck transformers for visual recognition. arXiv preprint arXiv:2101.11605, 2021.