Abstract

Convolutional Neural Networks (ConvNets) are commonly developed at a fixed resource budget, and then scaled up for better accuracy if more resources are available. In this paper, we systematically study model scaling and identify that carefully balancing network depth, width, and resolution can lead to better performance. Based on this observation, we propose a new scaling method that uniformly scales all dimensions of depth/width/resolution using a simple yet highly effective *compound coefficient*. We demonstrate the effectiveness of this method on scaling up MobileNets and ResNet.

To go even further, we use neural architecture search to design a new baseline network and scale it up to obtain a family of models, called *EfficientNets*, which achieve much better accuracy and efficiency than previous ConvNets. In particular, our EfficientNet-B7 achieves state-of-the-art 84.4% top-1 / 97.1% top-5 accuracy on ImageNet, while being **8.4x smaller** and **6.1x faster** on inference than the best existing ConvNet. Our EfficientNets also transfer well and achieve state-of-the-art accuracy on CIFAR-100 (91.7%), Flowers (98.8%), and 3 other transfer learning datasets, with an order of magnitude fewer parameters. Source code is at https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet.

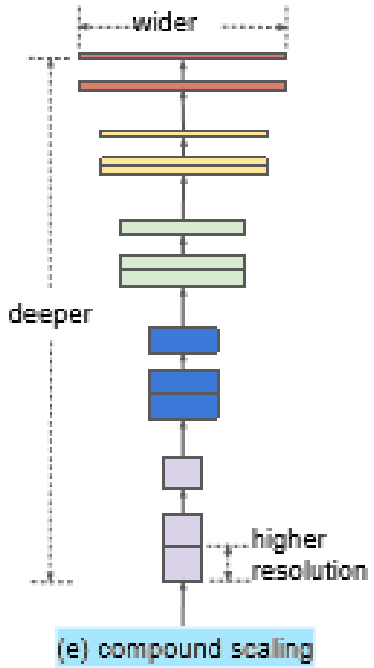


Table 1. **EfficientNet-B0 baseline network** – Each row describes a stage i with $\hat{L}_i$ layers, with input resolution $\langle \hat{H}_i, \hat{W}_i \rangle$ and output channels $\hat{C}_i$. Notations are adopted from equation 2.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
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 & FC	7 × 7	1280	1

- CNN Architecture

Purpose: Improving MobileNet, ResNet.

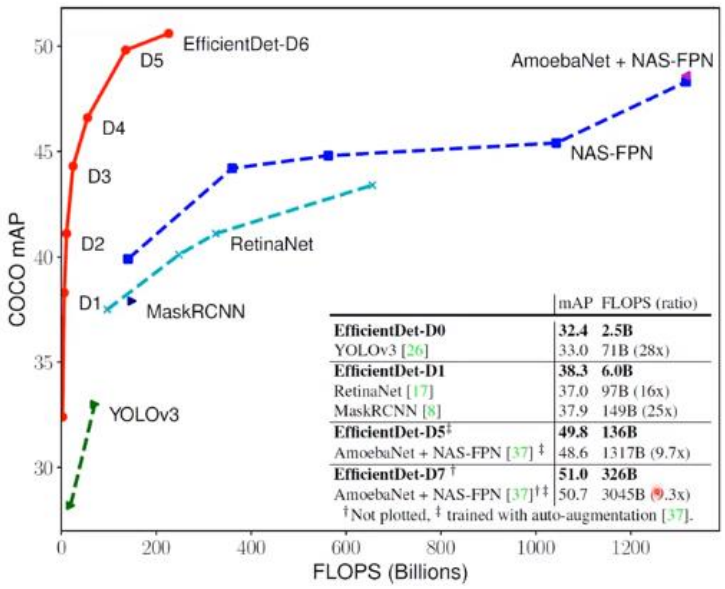
Method: proposing a new compound scaling method

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

Problem: it is common to scale only one of the three dimensions-depth, width, image size. But 세가지 방법을 조합해서 모델을 확장하는 경우나 적절한 조합을 찾는 방법에 대해서 명확하지 않음.

Suggestion: proposing **a new scaling method that uniformly scales all dimensions of depth/width/resolution** using a simple yet highly effective compound coefficient

==========1장정리==========



	mAP	FLOPS (ratio)
EfficientDet-D0	**32.4**	**2.5B**
YOLOv3 [26]	33.0	71B (28x)
EfficientDet-D1	**38.3**	**6.0B**
RetinaNet [17]	37.0	97B (16x)
MaskRCNN [8]	37.9	149B (25x)
EfficientDet-D5‡	**49.8**	**136B**
AmoebaNet + NAS-FPN [37] ‡	48.6	1317B (9.7x)
EfficientDet-D7 †	**51.0**	**326B**
AmoebaNet + NAS-FPN [37]†‡	50.7	3045B (9.3x)

†Not plotted, ‡ trained with auto-augmentation [37].

- EfficientDet-D7보면 연산량은 1/10이면서 정확성은 잘나온다

Intro.

- Real-World Resource에서는 small range of resource requirment에 focus

A Natural Question

- 과연 동시에 높은 정확성과 효율성을 만족하는가? -> YES!

Two Challenges(위 질문을 만족하기위한)

1. Efficient multi-scale feature fusion
   A. Most previous work simply sum them up without distinction
   B. However, they usually contribute to the fused output feature unequally(단순히 sum을 통해서 ouput을 추출하기엔 부족하다)
2. Model Scaling
   A. EfficientNet에서 한 것처럼 다양한 방식으로 compound scaling을 해주어야 한다.

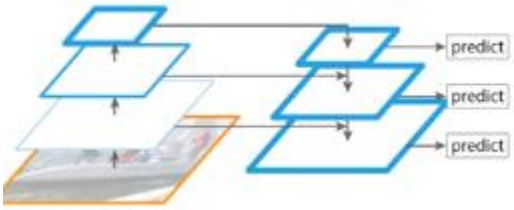
Propose: Combining EfficientNet backbones with BiFPN and compound scaling -> EfficientDet

Contribution

- Propose BiFPN(a weighted bidirectional feature network)
- A new compound scaling method
- Develop EfficientDet, a new family of one-stage detectors

- BiFPN – Problem Formulation
  - FPN
    - 각 feature pyramid network에 사용할 feature map을 P_in 이라 할때

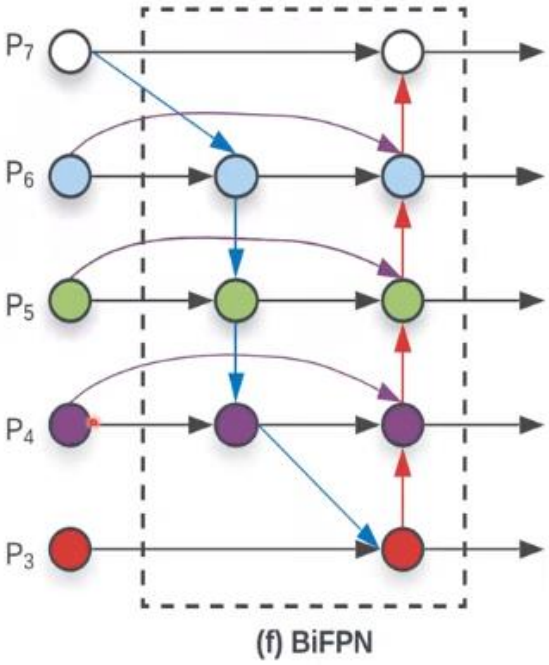
• Formaly, given a list of multi-scale features $\vec{P}^{in} = (P_{l_1}^{in}, P_{l_2}^{in}, \dots)$, where $P_{l_i}^{in}$ represents the feature at level l_i, the goal is to find a transformation f that can effectively aggregate different feature and output a list of new features: $\vec{P}^{out} = f(\vec{P}^{in})$



  - NAS-FPN
    - P3~P7을 조합해서 만들어보자
  - PANet
    - 기존 FPN은 top bottom구조인데 이는 bottom up으로 가는 것도 만들자(아래쪽에서 뽑아온 feature map에 서 작

은 물체를 가져올 수 있다)

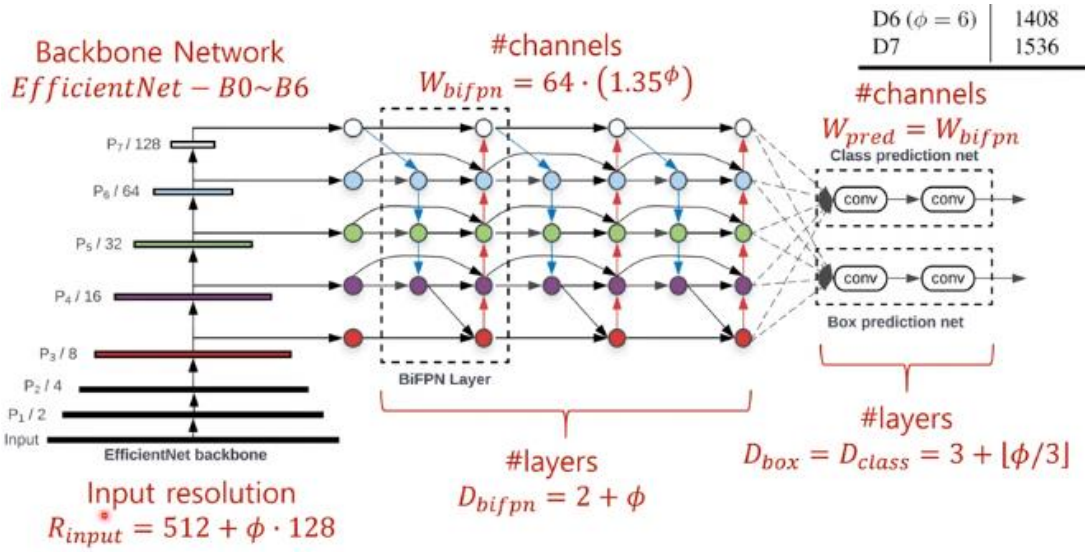
- BiFPN



(f) BiFPN

  - Weighted Feature Fusion
    - Unbounded Fusion
      - unbounded해서 값이 튀면 학습에 영향 줄 수 있다
    - Softmax-Based Fusion
      - gpu에서 돌려보면 속도를 많이 떨어뜨리는 요인이 됨
    - Fast Normalized Fusion
      - 실제로 BiFPN에서 도입한 개념

EfficientDet Architecture



FLOPs by $(\alpha \cdot \beta^2 \cdot \gamma^2)^{\phi}$

A simple compound coefficient pi to jointly scale up all dimensions of backbone network, BiFPN network, class/box network, resolution(이를 같은 비율로 키우는 것 compound)

하지만 object detection에서는 이 모두를 동시에 scaling dimension하기 힘들기 때문에, 어쩔 수 없이 heuristic-based scaling approach를 사용하게 됨

➔ Backbone network

Sample width/depth scaling coefficient of EfficientNet-B0 to B6

➔ BiFPN network

Wbifpn = 64·(1.35φ), Dbifpn = 3 + φ (

➔ Box/Class prediction network

Dbox = Dclass = 3 + [bφ/3c]

➔ Input image resolution

Rinput = 512 + φ·128

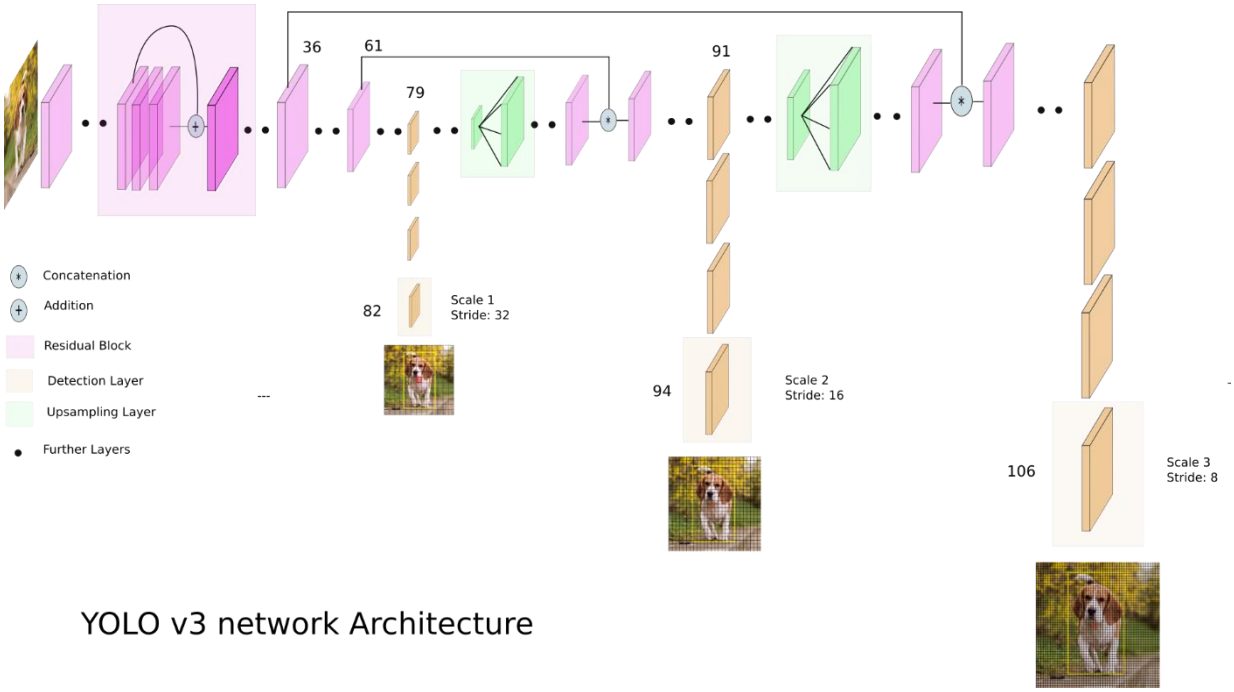
Experiments

EfficientDet-D0는 YOLOv3에 비해 성능이 조금 떨어졌지만, 연산량이 매우 작음

Conclusion

Weighted bidirectional feature network와 customized compound scaling method를 제안함(for improving both accuracy and efficiency)

EfficientDet-D7 achieves state-of-the-art를 달성하고, 4*정도 작고, 9.3배정도 작은 연산량을 가짐, 3.2배 빠르고(GPU) 8.1배 빠름CPU)



YOLO v3 network Architecture