Group 3

Manasi Rajiv Weginwar

Divya Gupta

Introduction:

As computer vision is continuously advancing, object detection has gotten progressively significant in situations that request high precision, however, have restricted computations, for example, robotics and driverless vehicles. Unfortunately, many current high-accuracy detectors do not fit these constraints. More importantly, real-world applications of object detection are run on a variety of platforms, which often demand different resources. A natural question, then, is how to design accurate and efficient object detectors that can also adapt to a wide range of resource constraints?

In "EfficientDet: Scalable and Efficient Object Detection", present another group of adaptable and effective object detectors. Expanding upon past work on scaling neural network (EfficientNet) and incorporating a novel bi-directional feature network (BiFPN) and new scaling rules, EfficientDet accomplishes cutting edge detection while being up to 9x smaller and utilizing altogether less computations contrasted with earlier best in class detectors.

Key Components:

- Backbone: The more advanced EfficientNets are used in backbone networks.
- BiFPN: BiFPN is being proposed as a new bi-directional feature network to enable easy and fast feature fusion.
- Scaling: Proposed use of a single compound scaling factor to govern the network depth, width, and resolution for all backbone, feature network, and prediction networks.

Architecture of the EfficientDet model-

EfficientDets are a family of object detection models, which achieve state-of-the-art 55.1mAP on COCO test-dev yet being 4x - 9x smaller and using 13x - 42x fewer FLOPs than previous detectors. These models also run 2x - 4x faster on GPU, and 5x - 11x faster on CPU than other detectors.

EfficientDets are developed based on the advanced backbone, a new BiFPN, and a new scaling technique:

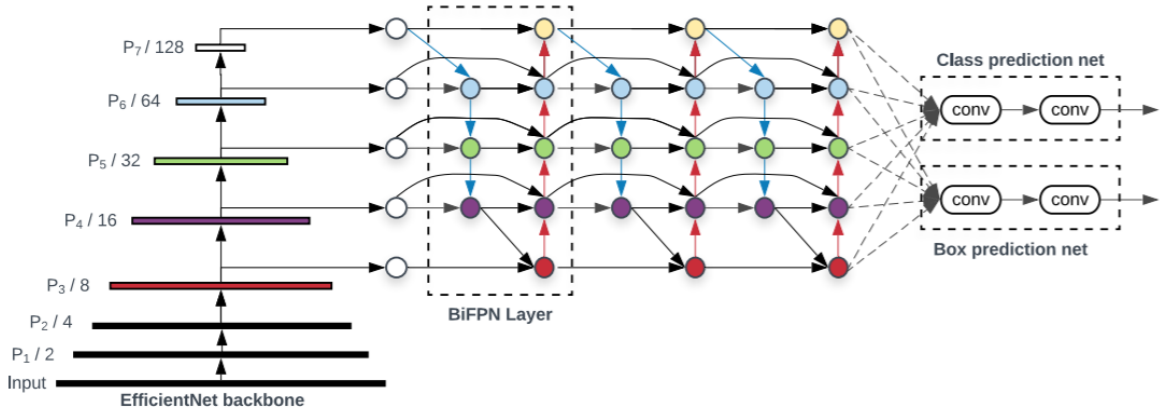

Figure 3: **EfficientDet architecture –** It employs EfficientNet [39] as the backbone network, BiFPN as the feature network, and shared class/box prediction network. Both BiFPN layers and class/box net layers are repeated multiple times based on different resource constraints as shown in Table 1.

A new compound scaling method was used for EfficientDet.

Compound Scaling: This is a new scaling method for object detection, which uses a simple compound coefficient ϕ to jointly scale up all dimensions of backbone, BiFPN, class/box network, and resolution.

Backbone: EfficientNets were used for this as the backbone networks. In this they reused the same width/depth scaling coefficients of EfficientNet-B0 to B6 such that they could easily reuse their ImageNet-pretrained checkpoints.

BiFPN: BiFPN, a bi-directional feature network enhanced with fast normalization, which enables easy and fast feature fusion:

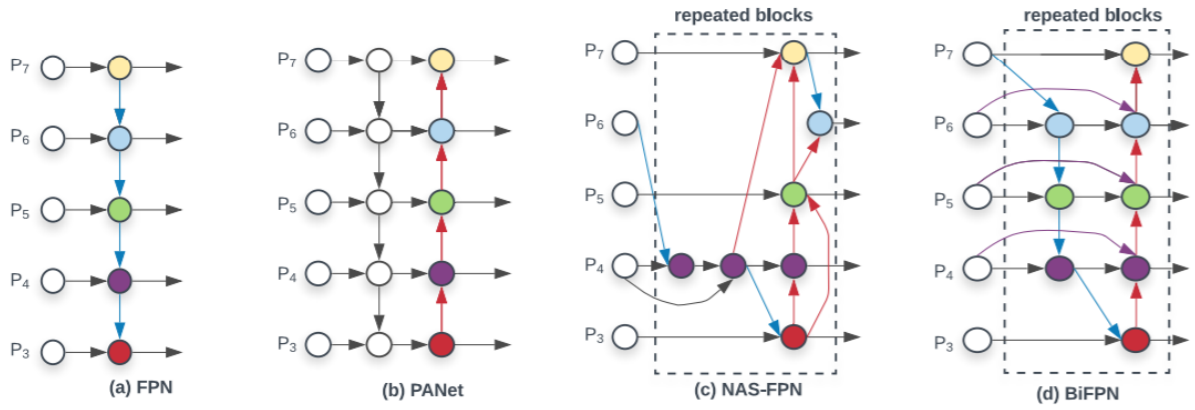

Figure 2: **Feature network design –** (a) FPN [23] introduces a top-down pathway to fuse multi-scale features from level 3 to 7 (P_3 - P_7); (b) PANet [26] adds an additional bottom-up pathway on top of FPN; (c) NAS-FPN [10] use neural architecture search to find an irregular feature network topology and then repeatedly apply the same block; (d) is our BiFPN with better accuracy and efficiency trade-offs.

Formally, BiFPN width and depth are scaled with the following equation:

$$W_{bifpn} = 64 \cdot \left(1.35^{\phi}\right), \qquad D_{bifpn} = 3 + \phi \qquad (1)$$

Scaling: A single compound scaling factor was used to govern the depth, width, and resolution for all backbone, feature & prediction networks. Scaling configurations for EfficientDet D0-D6:

	Input size R_{input}	Backbone Network	BiFPN #channels W_{bifpn}	BiFPN #layers D_{bifpn}	Box/class #layers D_{class}
D0 ($\phi = 0$)	512	B0	64	3	3
D1 ($\phi = 1$)	640	B1	88	4	3
D2 ($\phi = 2$)	768	B2	112	5	3
D3 ($\phi = 3$)	896	B3	160	6	4
D4 ($\phi = 4$)	1024	B4	224	7	4
D5 ($\phi = 5$)	1280	B5	288	7	4
D6 ($\phi = 6$)	1280	B6	384	8	5
D7 ($\phi = 7$)	1536	B6	384	8	5
D7x	1536	B7	384	8	5

Table 1: **Scaling configs for EfficientDet D0-D6 —** ϕ is the compound coefficient that controls all other scaling dimensions; *BiFPN, box/class net, and input size* are scaled up using equation 1, 2, 3 respectively.

Box/class prediction network: Fixed their width to be always the same as BiFPN (i.e., $W_{pred} = W_{bifpn}$), but linearly increase the depth (#layers) using equation:

$$D_{box} = D_{class} = 3 + \lfloor \phi/3 \rfloor \qquad (2)$$

Input image resolution: Since feature level 3-7 are used in BiFPN, the input resolution must be dividable by $2^7 = 128$, so they linearly increase resolutions using equation:

$$R_{input} = 512 + \phi \cdot 128 \qquad (3)$$

Following Equations 1,2,3 with different ϕ, they have developed EfficientDet-D0 (ϕ = 0) to D7 (ϕ = 7) as shown in Table 1, where D7 and D7x have the same BiFPN and head, but D7 uses higher resolution and D7x uses larger backbone network and one more feature level (from P3 to P8).

This model family starts from EfficientDet-D0, which has comparable accuracy as YOLOv3. Then they scale up this baseline model using compound scaling method to obtain a list of detection models EfficientDet-D1 to D6, with different trade-offs between accuracy and model complexity.

Specifications-

- Researchers of these models trained these models with batch size 128 on 32 TPUv3 chips and the results were as follows:

Model	test-dev AP	test-dev AP_{50}	test-dev AP_{75}	val AP	Params	Ratio	FLOPs	Ratio	Latency (ms) TitianV	Latency (ms) V100
EfficientDet-D0 (512)	**34.6**	**53.0**	**37.1**	**34.3**	**3.9M**	**1x**	**2.5B**	**1x**	**12**	**10.2**
YOLOv3 [34]	33.0	57.9	34.4	-	-	-	71B	28x	-	-
EfficientDet-D1 (640)	**40.5**	**59.1**	**43.7**	**40.2**	**6.6M**	**1x**	**6.1B**	**1x**	**16**	**13.5**
RetinaNet-R50 (640) [24]	39.2	58.0	42.3	39.2	34M	6.7x	97B	16x	25	-
RetinaNet-R101 (640)[24]	39.9	58.5	43.0	39.8	53M	8.0x	127B	21x	32	-
EfficientDet-D2 (768)	**43.9**	**62.7**	**47.6**	**43.5**	**8.1M**	**1x**	**11B**	**1x**	**23**	**17.7**
Detectron2 Mask R-CNN R101-FPN [1]	-	-	-	42.9	63M	7.7x	164B	15x	-	56‡
Detectron2 Mask R-CNN X101-FPN [1]	-	-	-	44.3	107M	13x	277B	25x	-	103‡
EfficientDet-D3 (896)	**47.2**	**65.9**	**51.2**	**46.8**	**12M**	**1x**	**25B**	**1x**	**37**	**29.0**
ResNet-50 + NAS-FPN (1024) [10]	44.2	-	-	-	60M	5.1x	360B	15x	64	-
ResNet-50 + NAS-FPN (1280) [10]	44.8	-	-	-	60M	5.1x	563B	23x	99	-
ResNet-50 + NAS-FPN (1280@384)[10]	45.4	-	-	-	104M	8.7x	1043B	42x	150	-
EfficientDet-D4 (1024)	**49.7**	**68.4**	**53.9**	**49.3**	**21M**	**1x**	**55B**	**1x**	**65**	**42.8**
AmoebaNet+ NAS-FPN +AA(1280)[45]	-	-	-	48.6	185M	8.8x	1317B	24x	246	-
EfficientDet-D5 (1280)	**51.5**	**70.5**	**56.1**	**51.3**	**34M**	**1x**	**135B**	**1x**	**128**	**72.5**
Detectron2 Mask R-CNN X152 [1]	-	-	-	50.2	-	-	-	-	-	234‡
EfficientDet-D6 (1280)	**52.6**	**71.5**	**57.2**	**52.2**	**52M**	**1x**	**226B**	**1x**	**169**	**92.8**
AmoebaNet+ NAS-FPN +AA(1536)[45]	-	-	-	50.7	209M	4.0x	3045B	13x	489	-
EfficientDet-D7 (1536)	**53.7**	**72.4**	**58.4**	**53.4**	**52M**		**325B**		**232**	**122**
EfficientDet-D7x (1536)	**55.1**	**74.3**	**59.9**	**54.4**	**77M**		**410B**		**285**	**153**

We omit ensemble and test-time multi-scale results [30, 12]. RetinaNet APs are reproduced with our trainer and others are from papers.
‡Latency numbers with ‡ are from detectron2, and others are measured on the same machine (TensorFlow2.1 + CUDA10.1, no TensorRT).

Table 2: **EfficientDet performance on COCO** [25] – Results are for single-model single-scale. `test-dev` is the COCO test set and `val` is the validation set. `Params` and `FLOPs` denote the number of parameters and multiply-adds. `Latency` is for inference with batch size 1. `AA` denotes auto-augmentation [45]. We group models together if they have similar accuracy, and compare their model size, FLOPs, and latency in each group.

In our COVID project, we tried to implement this model along with Nvidia tesla V100 to train our Social Distancing module, requirements of Tesla V100 are as follows:

SYSTEM SPECIFICATIONS	
GPUs	**16X NVIDIA® Tesla V100**
GPU Memory	**512GB total**
Performance	**2 petaFLOPS**
NVIDIA CUDA® Cores	**81920**
NVIDIA Tensor Cores	**10240**
NVSwitches	**12**
Maximum Power Usage	**10 kW**
CPU	**Dual Intel Xeon Platinum 8168, 2.7 GHz, 24-cores**
System Memory	**1.5TB**
Network	**8X 100Gb/sec Infiniband/100GigE Dual 10/25Gb/sec Ethernet**
Storage	**OS: 2X 960GB NVME SSDs Internal Storage: 30TB (8X 3.84TB) NVME SSDs**
Software	**Ubuntu Linux OS See Software stack for details**
System Weight	**340 lbs (154.2 kgs)**
System Dimensions	**Height: 17.3 in (440.0 mm) Width: 19.0 in (482.3 mm) Length: 31.3 in (795.4 mm) - No Front Bezel 32.8 in (834.0 mm) - With Front Bezel**
Operating Temperature Range	**5°C to 35°C (41°F to 95°F)**

Applications:

1. As per the research done, COCO dataset, a widely used benchmark dataset for object detection. EfficientDet D4 is faster in performance as compared to previous EfficientNet.[1]
2. While the EfficientDet models are mainly designed for object detection, papers have also examined their performance on other tasks, such as semantic segmentation. And when the model is compared with prior state-of-the-art segmentation models for Pascal VOC 2012, a widely used dataset for segmentation benchmark. The accuracy and quality are more for EfficientDet.[2]

References-

1. EfficientDet: Scalable and Efficient Object Detection, Mingxing Tan Ruoming Pang Quoc V. Le Google Research, Brain Team, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (July 2020)

2. https://github.com/signatrix/efficientdet

3. https://github.com/google/automl/tree/master/efficientdet

4. https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/dgx-2/dgx-2-print-datasheet-738070-nvidia-a4-web-uk.pdf

5. https://en.wikipedia.org/wiki/Image_segmentation

6. https://ai.googleblog.com/2020/04/efficientdet-towards-scalable-and.html