Brief

- *Title:* EfficientDet: Scalable and Efficient Object Detection
- *Authors:* Mingxing Tan, Ruoming Pang, Quoc V. Le
- *Institution:* Google Research, Brain Team
- *Link:* https://arxiv.org/pdf/1911.09070.pdf
- *Code:* https://github.com/google/automl/tree/master/efficientdet
- *Key-words:* neural net architecture, computer vision, EfficientDet, feature extraction, object detection, CVPR 2020

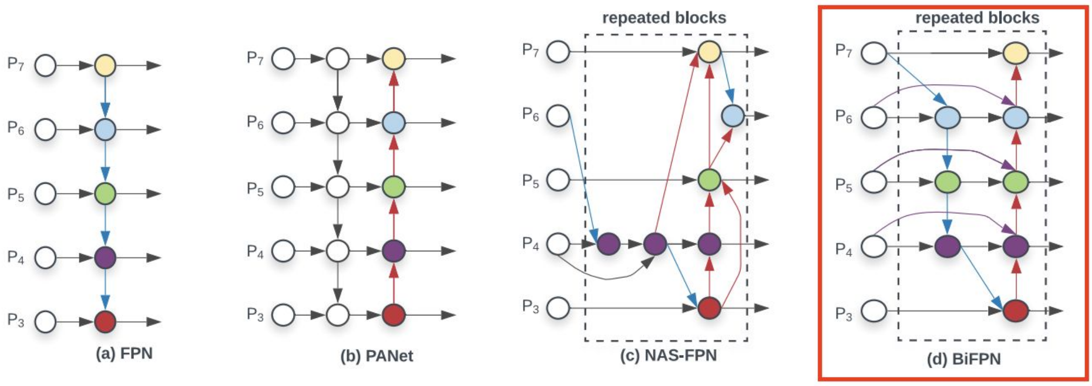
Summary

What:

- *The general idea* is to build a scalable model architecture with both higher accuracy and better efficiency across different resource constraints.
- Make it possible to use the model in different *real-world applications* like robotics or self-driving cars that need high accuracy object detectors but are limited in computational resources. Also, make it possible to use high accuracy models for applications with different resource constraints.
- Proposed:
    - **BiFPN**: efficient bidirectional cross-scale connections and weighted feature fusion.
    - **New compound scaling method**: uses the only coefficient φ to jointly scale-up all dimensions of backbone, BiFPN, class/box network, and image resolution.
- In numbers:
    - The presented model EfficientDet-D7x achieves a new state-of-the-art 55.1 AP, outperforming prior art in both accuracy (+4 AP) and efficiency (7x fewer FLOPs)
    - achieves better efficiency than previous detectors, being 4x–9x smaller and using 13x-42x fewer FLOPs across a wide range of accuracy or resource constraints. For example, compared to RetinaNet and Mask-RCNN, the EfficientDet-D1 achieves similar accuracy with up to 8x fewer parameters and 25x fewer FLOPS.

How:

1. **Proposed BiFPN**: bidirectional cross-scale connections and weighted feature fusion, which is an improved classical FPN approach.

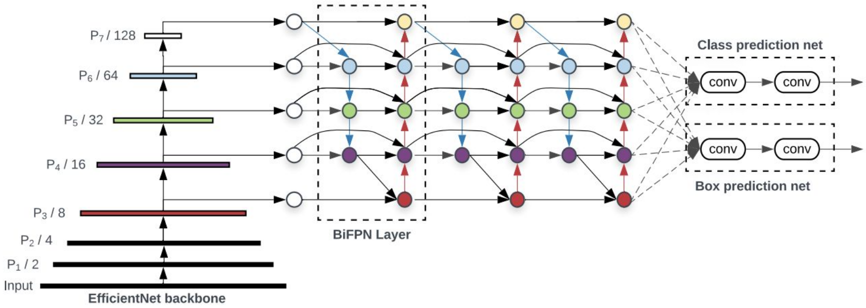


The major updates are:

- Added extra bottom-up path aggregation network like in PANet
- Performed several optimizations for cross-scale connections:
  - Remove nodes that have only one input edge as they have less impact on feature fusion, which leads to simplification of network
  - Add an extra edge from the same level original input to the output node in order to fuse more features
  - Consider each top-down and one bottom-up path as one bidirectional layer and repeat it multiple times
- Added input weighting solves the problem of inequality of contribution to the output of features from different resolutions. Also proposed three weighting strategies:
  - *Unbounded fusion*: $O = \sum_i w_i \cdot I_i$, where w_i is a learnable weight. However, scalar weight here is unbounded, so it could potentially cause training instability.
  - *Softmax-based fusion*: $O = \sum_i \frac{e^{w_i}}{\sum_j e^{w_j}}$. That solves the problem of probability normalization but leads to a significant slowdown
  - *Fast normalized fusion*: $O = \sum_i \frac{w_i}{\varepsilon + \sum_j w_j}$, where $w_i \geq 0$ because of applying Relu after each w_i. That is also probability normalization, but 30% faster on GPUs than softmax-based and with similar learning behavior and accuracy.

2. Present EfficientDet detection models family with **new compound scaling method.**

EfficientDet architecture



The architecture is:

- ImageNet-pretrained EfficientNets as the backbone network
- Feature network that takes level 3-7 features and applies BiFPN several times
- Class and box network that makes prediction respectively to the task.

The new compound scaling method uses the only one coefficient φ to simultaneously scale backbone, BiFPN, class/box network, and resolution with the following logic:

- *Backbone network*: reuse the same width/depth scaling coefficients of EfficientNet-B0 to B6, reusing ImageNet-pretrained checkpoints.
- *BiFPN network:*
  - width (#channels) are calibrated with $W_{bifpn} = 64 \cdot 1.35^{\varphi}$
  - depth (#layers) are figured with $D_{bifpn} = 3 + \varphi$
- *Box/class prediction network:*
  - width is always the same as BiFPN: $W_{pred} = W_{bifpn}$
  - depth (#layers) are changing by $D_{box} = D_{class} = 3 + \lfloor \varphi/3 \rfloor$
- *Input image resolution:* $R_{input} = 512 + \varphi \cdot 128$

Scaling configs 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

Results:

- Evaluated EfficientDet for Object Detection on COCO 2017 detection dataset. Models were trained with the same parameters on focal loss. D0-D6 models are trained for 300 epochs with a 128 batch size and D7/D7x for 600 epochs.

EfficientDet performance on COCO

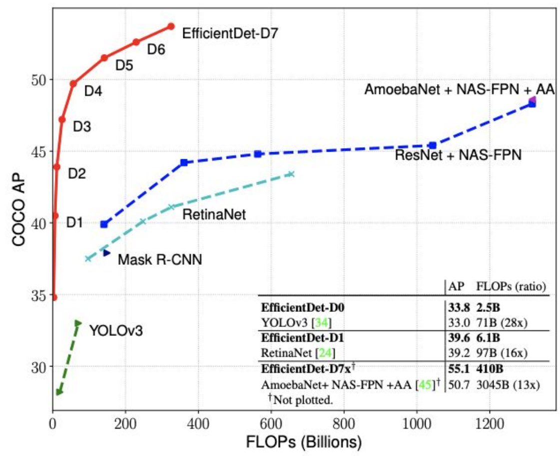
Model	test-dev AP	test-dev AP_{50}	test-dev AP_{75}	val AP	Params	Ratio	FLOPs	Ratio	Latency (ms) TitianV	Latency (ms) V100
EfficientDet-D0 (512)	**34.6**	**53.0**	**37.1**	**34.3**	**3.9M**	**1x**	**2.5B**	**1x**	**12**	**10.2**
YOLOv3	33.0	57.9	34.4	-	-	-	71B	28x	-	-
EfficientDet-D1 (640)	**40.5**	**59.1**	**43.7**	**40.2**	**6.6M**	**1x**	**6.1B**	**1x**	**16**	**13.5**
RetinaNet-R50 (640)	39.2	58.0	42.3	39.2	34M	6.7x	97B	16x	25	-
RetinaNet-R101 (640	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	-	-	-	42.9	63M	7.7x	164B	15x	-	56‡
Detectron2 Mask R-CNN X101-FPN	-	-	-	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)	44.2	-	-	-	60M	5.1x	360B	15x	64	-
ResNet-50 + NAS-FPN (1280)	44.8	-	-	-	60M	5.1x	563B	23x	99	-
ResNet-50 + NAS-FPN (1280@384)	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)	-	-	-	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	-	-	-	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)	-	-	-	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**

The comparison is made for different groups of models within various accuracy/performance constraints. As a result:

➢ EfficientDet having better scores in each group, being 4x - 9x smaller and using 13x - 42x fewer FLOPs

➢ EfficientDet-D7x achieves a new state-of-the-art 55.1 AP on test-dev, improving the score by +4 AP and using 7x fewer FLOPs at the same time

➢ EfficientDet models are up to 4.1x faster on GPU and 10.8x faster on CPU



- Checked the influence of backbone and a new BiFPN separately and showed that both crucial for final models. Both EfficientNetB3 and BiFPN improve accuracy and decrease the number of FLOPs.

Disentangling backbone and BiFPN

	AP	Parameters	FLOPs
ResNet50 + FPN	37.0	34M	97B
EfficientNet-B3 + FPN	40.3	21M	75B
EfficientNet-B3 + **BiFPN**	44.4	12M	24B

- Evaluated EfficientDet for Semantic Segmentation on Pascal VOC. For experiment used only one configuration of EfficientDet-D4 based model with BiFPN that uses levels 2-7, and P2 is used for final per-pixel classification. EfficientDet achieves 1.7% better accuracy with 9.8x fewer FLOPs than the prior art of DeepLabV3+.