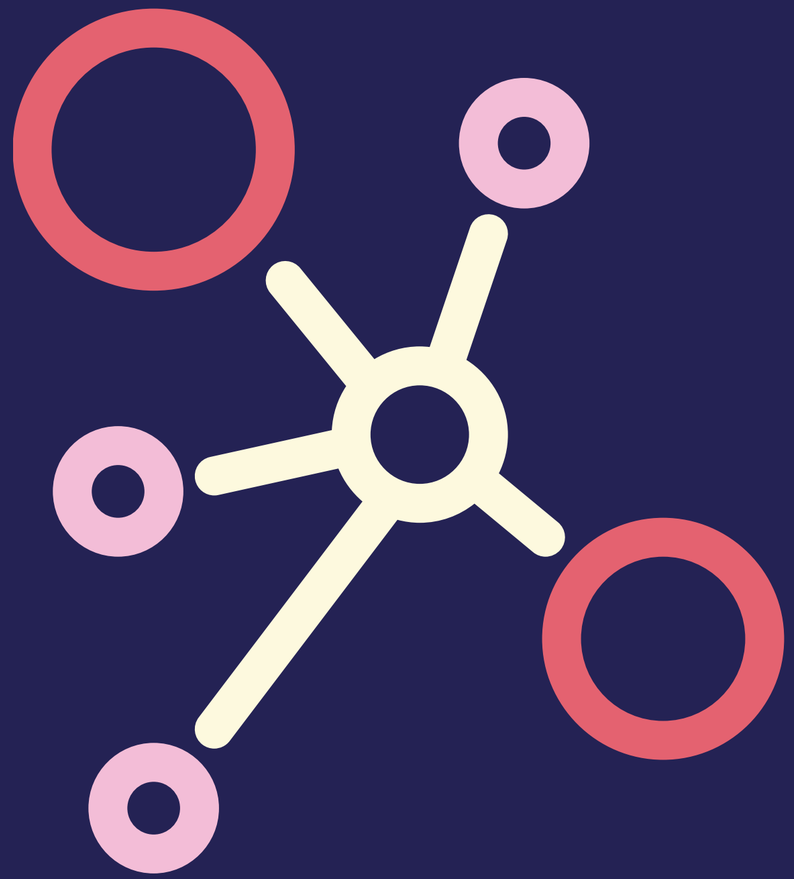


Mobile Net

Efficient Convolutional Neural Networks for Mobile Vision Applications

Talya Meizlish

Esther Binnes

28 June 2023

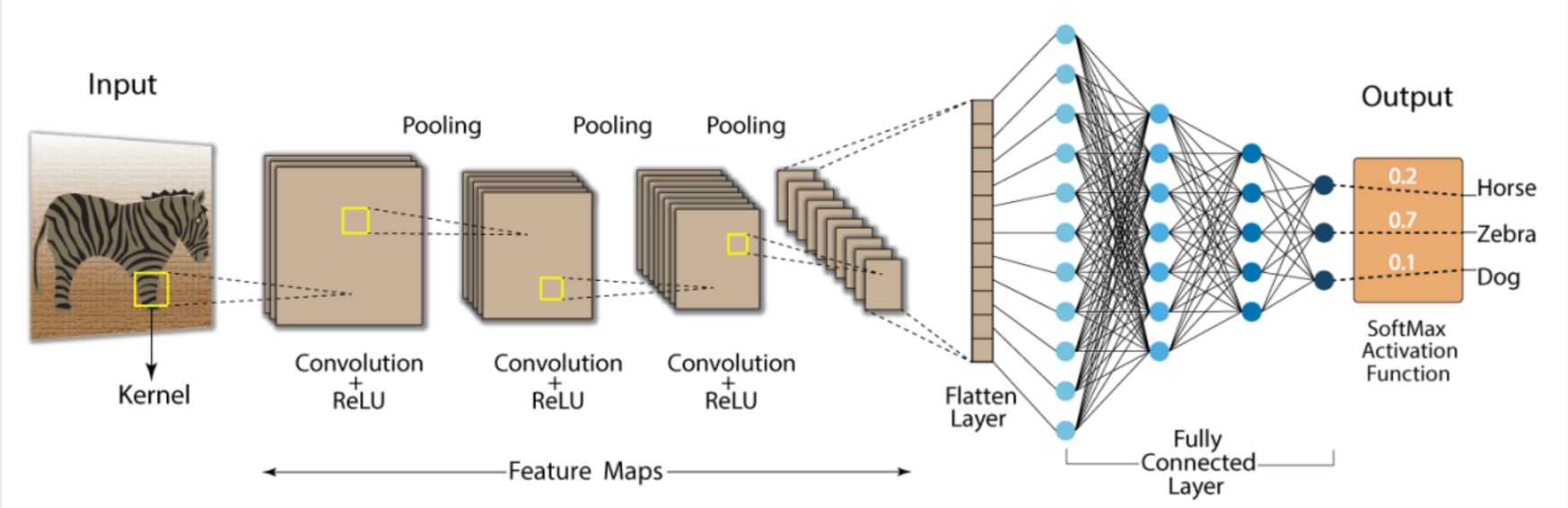
Terms

Convolutional Neural Network

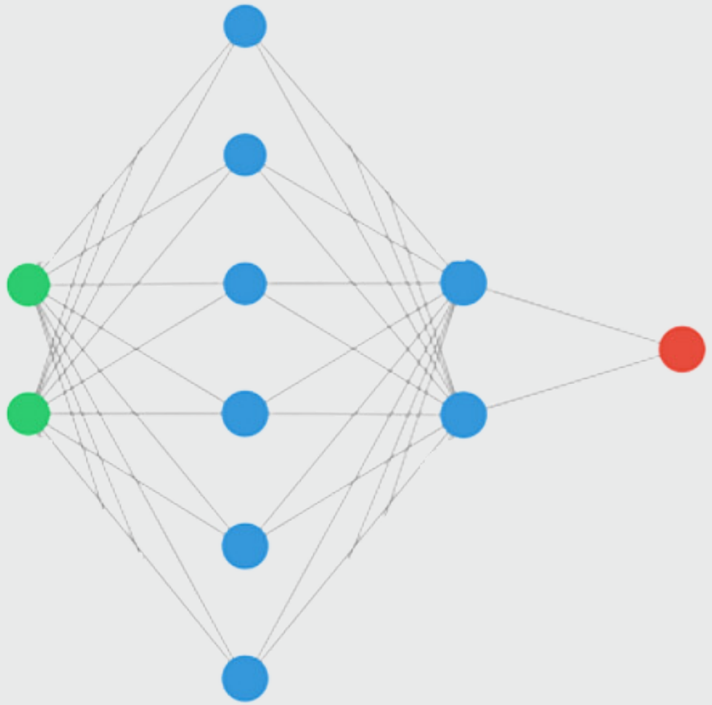
A CNN is specialized type of neural network commonly used for image classificationand object detection.

CNNs are composed of multiple layers, including convolutional layers, pooling layers, and fully connected layers.

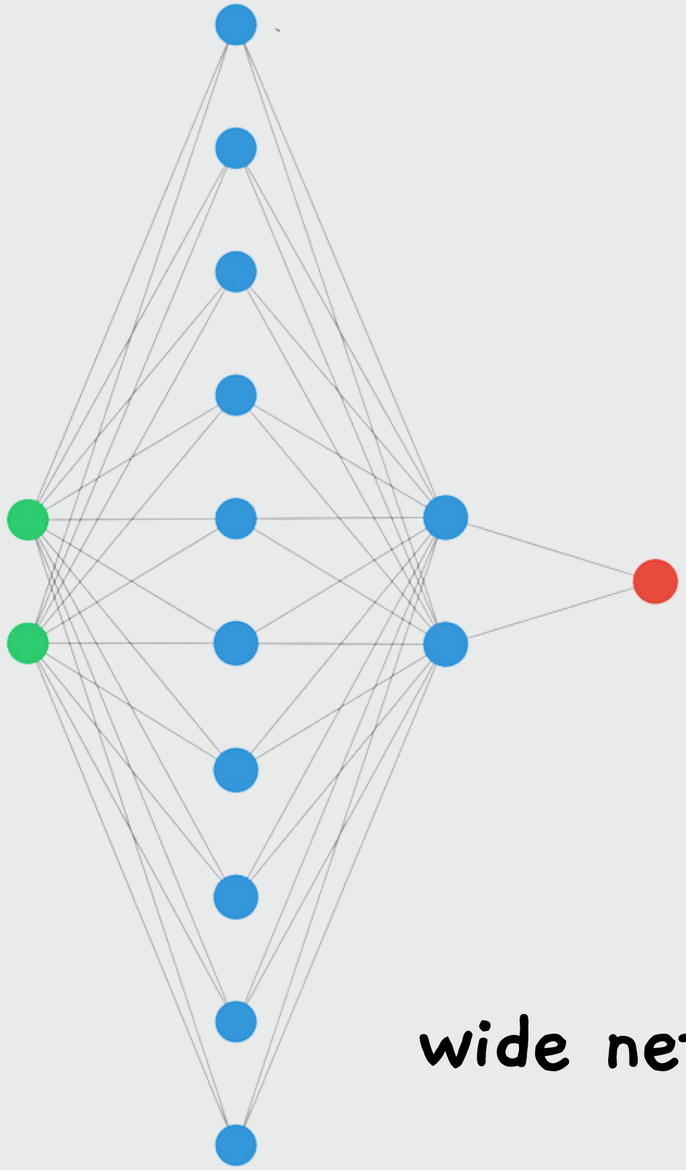
Convolutional Layers: These layers consist of multiple learnable filters (also called kernels) that slide over the input image. Each filter performs a convolution operation by element-wise multiplying its weights with a small region of the image and summing the results. The convolution operation helps capture local patterns and spatial features, such as edges and textures.



Width of Neural Network

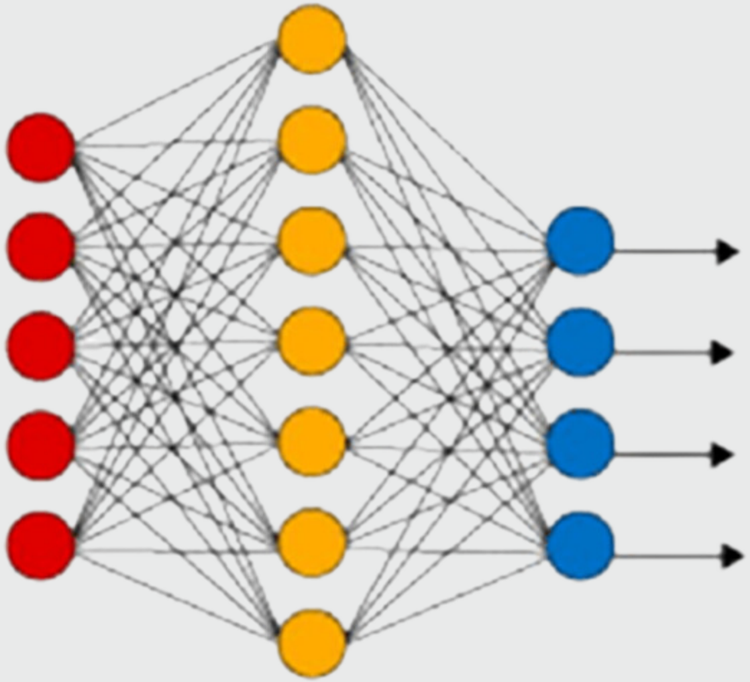


narrow network

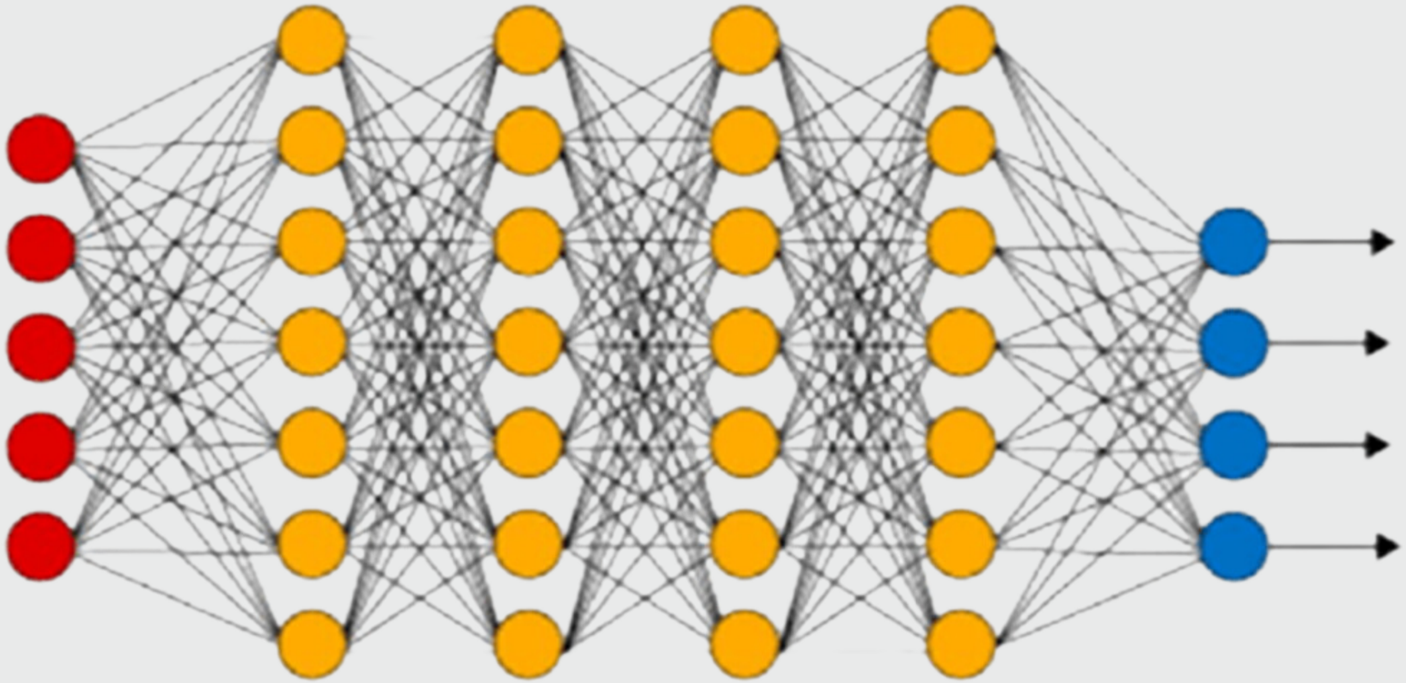


wide network

Depth of Neural Network

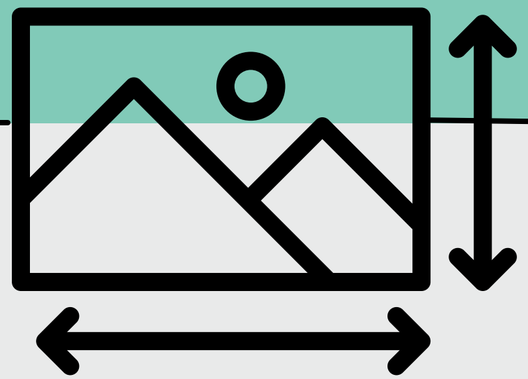


shallow network



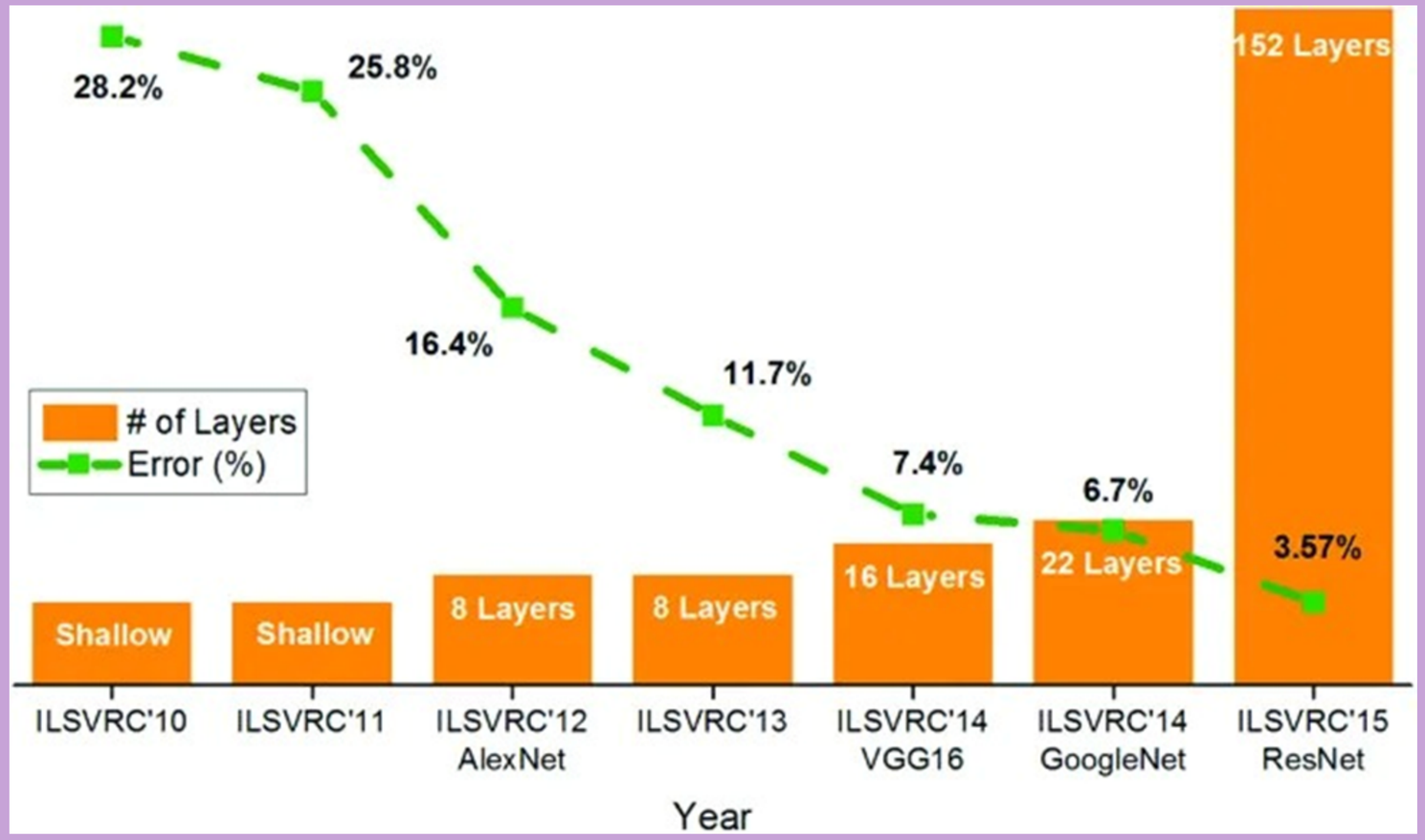
deep network

Image Resolution



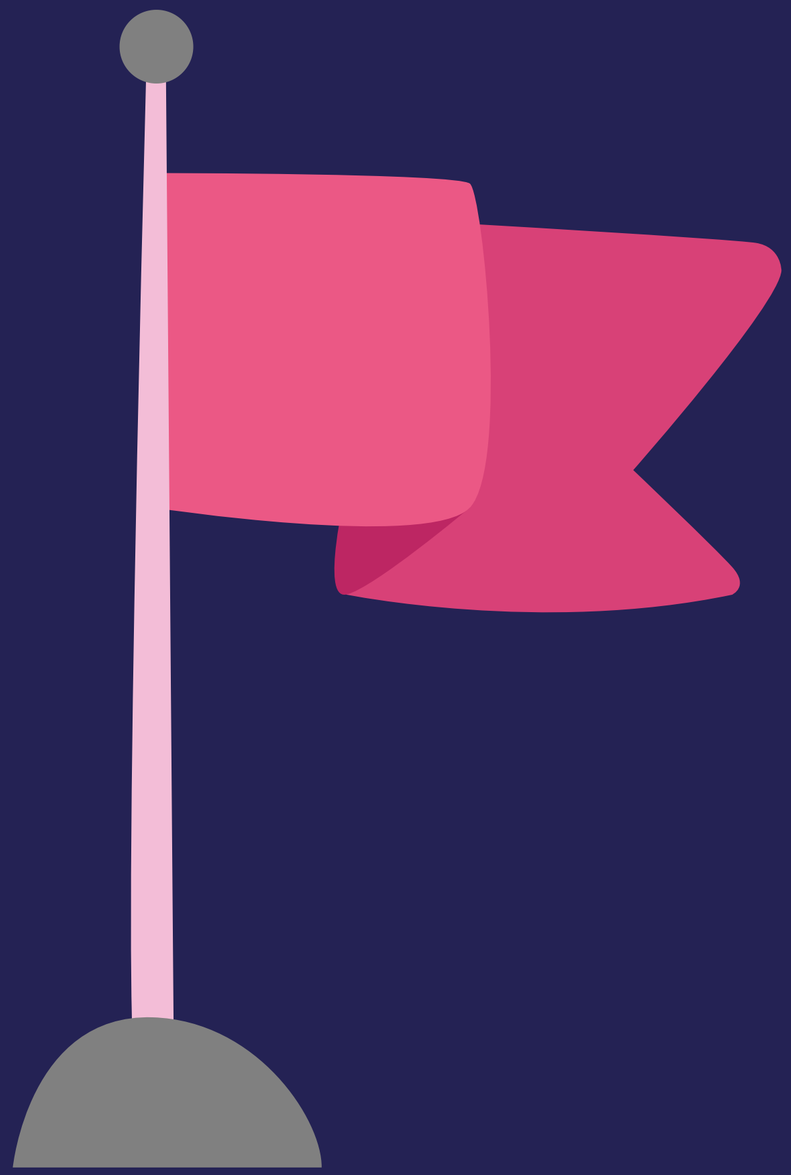
Deep & Wide Neural Network

multiple hidden layers lead to any function, which enable better efficiency in learning the relationship between input and output.



The Problem

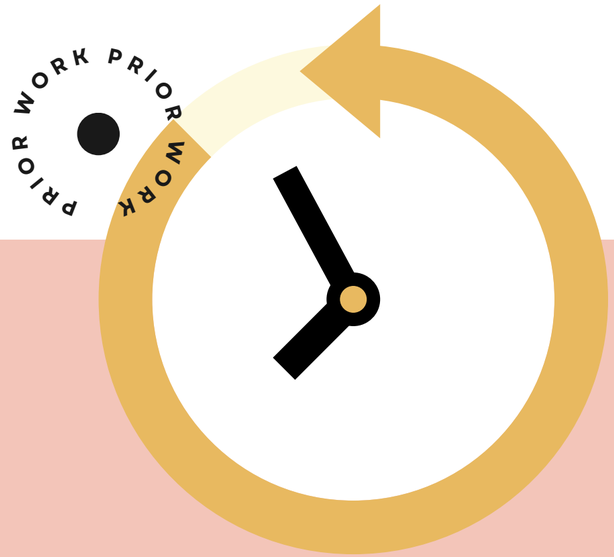
The complexity of a large neural network can be significant



The Goal

Build small models with low latency and minimal damage to accuracy

Prior Work



01.

Pretrained models

Starting point for transfer learning. Rather than training a model from scratch, the knowledge learned by a pretrained model on a large-scale dataset is fine-tuned on a smaller, task-specific dataset. the model can learn to perform well on a specific task with less training data and computational resources.

02.

Knowledge Distillation

Using a larger network to teach a smaller network.
Training a deep network and a shallow network, and try to force the success of the deep network on the shallow one.
In training, you run both networks and in testing only the shallow network.

Disadvantages

The networks on which they are based are heavy, therefore the need for a small change will entail an operation with too much weight

Mobile Net

MobileNet Architecture

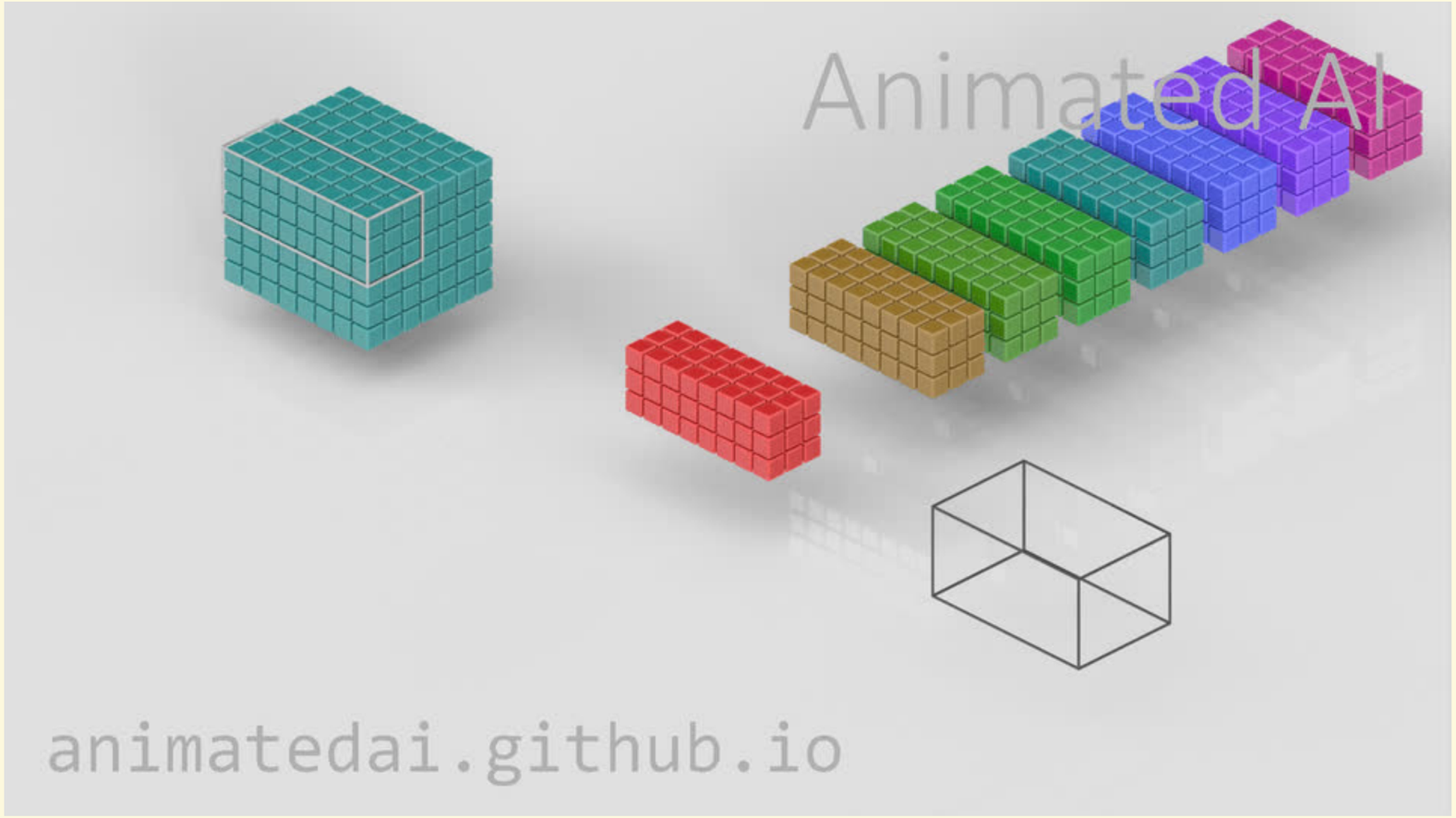
1. Depthwise Separable Convolution

2. Width Multiplier: Thinner Models

3. Resolution Multiplier: Reduced Representation

DEPTHWISE SEPARABLE CONVOLUTION

Standrd Convolution



computational cost: $D_f \cdot D_f \cdot M \cdot D_k \cdot D_k \cdot N$

- $D_f \cdot D_f$: Input feature map F
- M: Number of input channels
- $D_k \cdot D_k$: Kernel dimensions
- N: Number of output channels

DEPTHWISE SEPARABLE CONVOLUTION

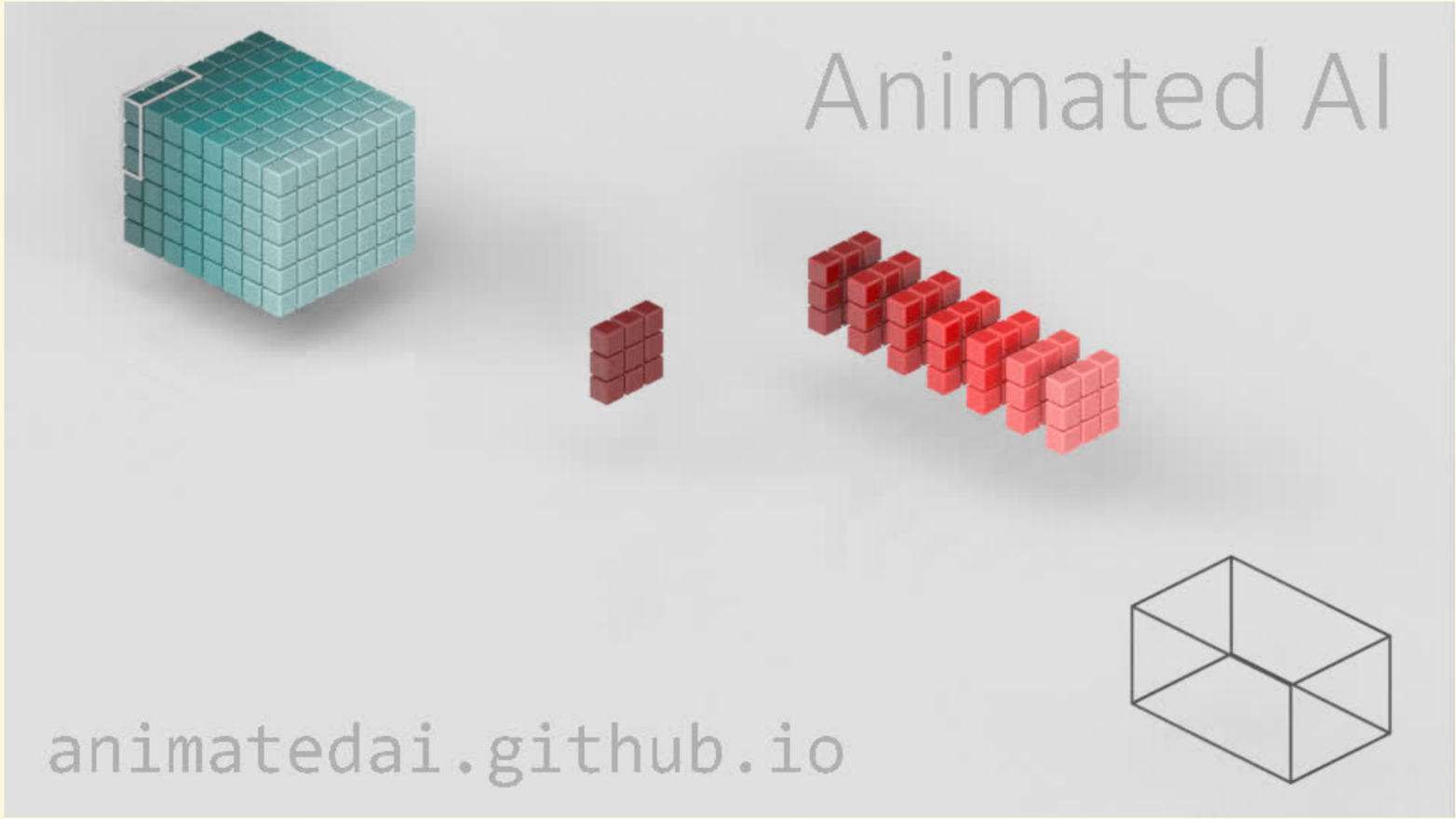
Depthwise Separable Convolution

2 Pointwise convolution

1 Depthwise convolution

DEPTHWISE SEPARABLE CONVOLUTION

Depthwise Convolution



computational cost: $D_f \cdot D_f \cdot M \cdot D_k \cdot D_k$

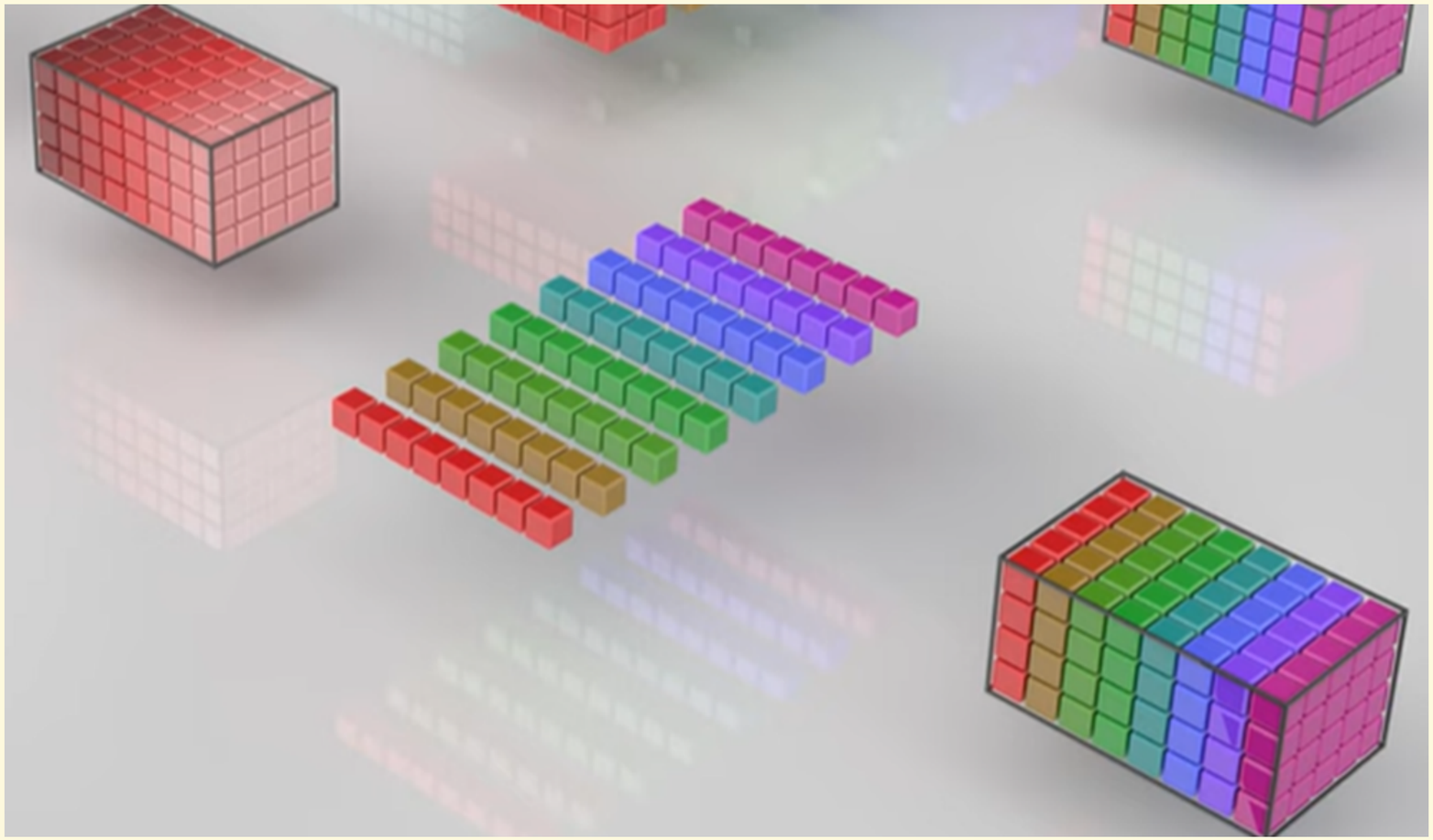
Input feature map F

Number of input channels

Kernel dimensions

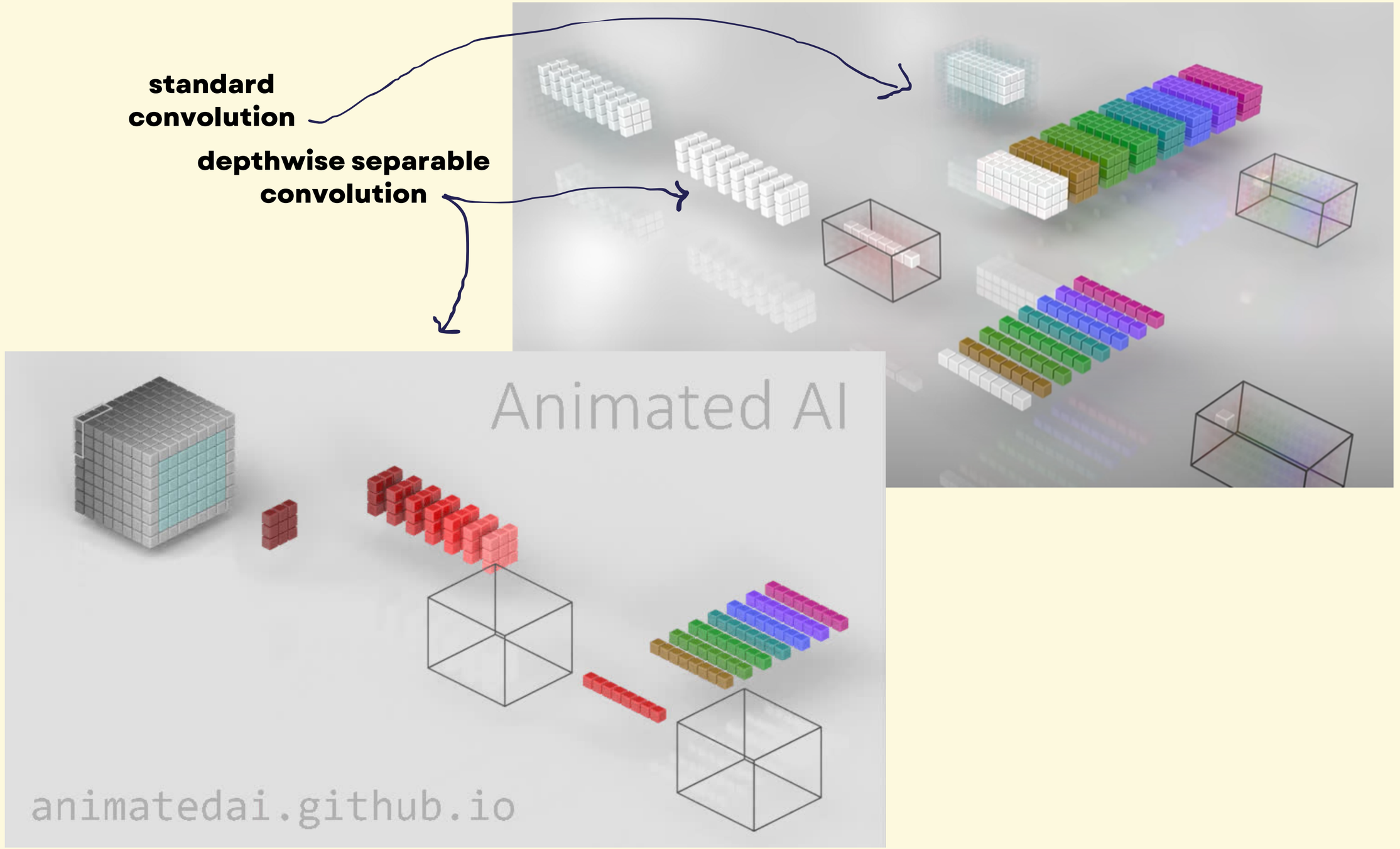
DEPTHWISE SEPARABLE CONVOLUTION

Pointwise Convolution

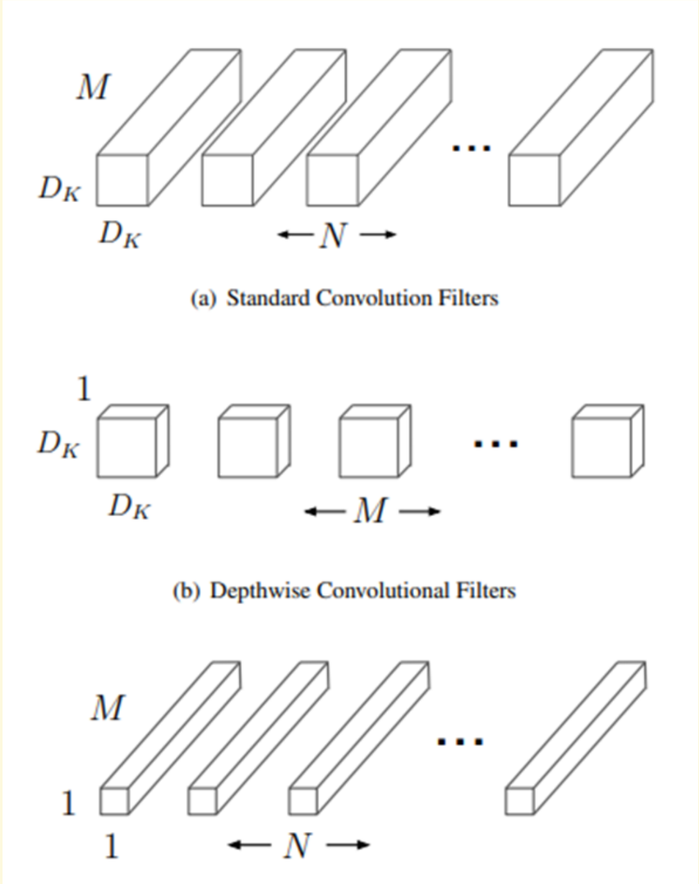


computational cost: $\underbrace{D_f \cdot D_f}_{\text{Input feature map F}} \cdot \underbrace{M}_{\substack{\text{Number of} \\ \text{input} \\ \text{channels}}} \cdot \underbrace{N}_{\substack{\text{Number of} \\ \text{output} \\ \text{channels}}}$

DEPTHWISE SEPARABLE CONVOLUTION



Depthwise Separable Convolution



Depthwise Convolution + Pointwise Convolution

=

Depthwise Separable Convolution

=

Standard Convolution with fewer calculation operations

Standard Convolution:

$$D_f \cdot D_f \cdot M \cdot D_k \cdot D_k \cdot N$$

Depthwise Separable Convolution:

$$\underbrace{D_f \cdot D_f \cdot M \cdot D_k \cdot D_k}_{\text{Depthwise Convolution}} + \underbrace{D_f \cdot D_f \cdot M \cdot N}_{\text{Pointwise Convolution}}$$

$$=$$

$$D_f \cdot D_f \cdot M \cdot (D_k \cdot D_k + N)$$

$$D_f \cdot D_f \cdot M \cdot D_k \cdot D_k \cdot N > D_f \cdot D_f \cdot M(D_k \cdot D_k + N)$$

$$D_k \cdot D_k \cdot N > D_k \cdot D_k + N$$

Network Structure

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

The first layer is a full convolution

All layers are depthwise separable convolutions

All layers are followed by a batchnorm and ReLU nonlinearity

The final fully connected layer has no nonlinearity and feeds into a softmax layer for classification

Width Multiplier: Thinner Models

- The number of input channels M becomes αM
- The number of output channels N becomes αN.
- Typical settings: 1, 0.75, 0.5, 0.25.
- α = 1 is the baseline MobileNet and α < 1 are reduced MobileNets.
- Reduces computational cost and number of parameters quadratically by roughly α^2 .

$$\alpha \in (0, 1] \qquad D_K \cdot D_K \cdot \alpha M \cdot D_F \cdot D_F + \alpha M \cdot \alpha N \cdot D_F \cdot D_F$$

WIDTH MULTIPLIER: THINNER MODELS

ImageNet:

14,000,000 images

20,000 classes

Table 6. MobileNet Width Multiplier

Width Multiplier	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
0.75 MobileNet-224	68.4%	325	2.6
0.5 MobileNet-224	63.7%	149	1.3
0.25 MobileNet-224	50.6%	41	0.5

Resolution Multiplier:

Reduced Representation

- Is set implicitly by setting the input resolution
- Typical settings: 224, 192, 160, 128.
- ρ = 1 is the baseline MobileNet and ρ < 1 are reduced MobileNets.
- Reduces computational cost quadratically by roughly ρ^2 .

$$\rho \in (0, 1] \quad D_K \cdot D_K \cdot \alpha M \cdot \rho D_F \cdot \rho D_F + \alpha M \cdot \alpha N \cdot \rho D_F \cdot \rho D_F$$

RESOLUTION MULTIPLIER: REDUCED REPRESENTATION

ImageNet:

14,000,000 images

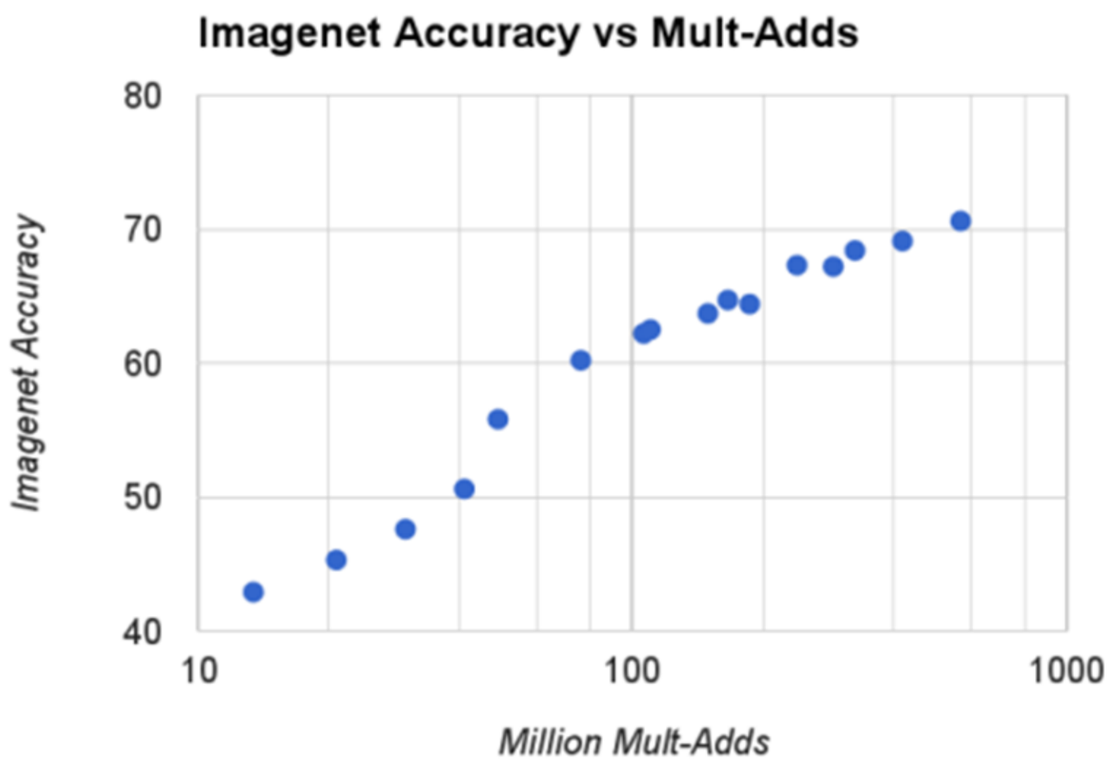
20,000 classes

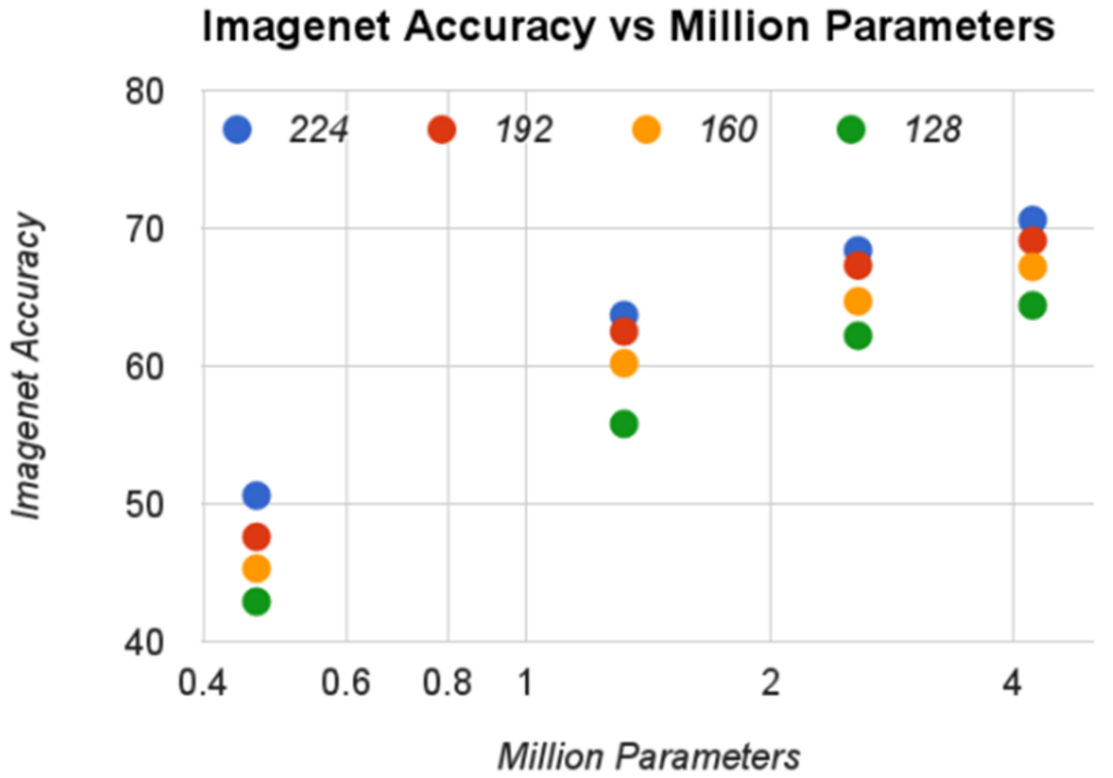
Table 7. MobileNet Resolution

Resolution	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
1.0 MobileNet-192	69.1%	418	4.2
1.0 MobileNet-160	67.2%	290	4.2
1.0 MobileNet-128	64.4%	186	4.2

Experiments

Experiments





$\alpha \in \{1, 0.75, 0.5, 0.25\}$

resolutions $\{224, 192, 160, 128\}$

Running MobileNet on ImageNet data

Experiments

full MobileNet VS GoogleNet and VGG16

Table 8. MobileNet Comparison to Popular Models

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

MobileNet is nearly as accurate as VGG16 while being 32 times smaller and 27 times less compute intensive.
MobileNet is more accurate than GoogleNet while being 1.6 times smaller and 2.7 times less computation

Experiments

reduced MobileNet VS Squeezenet and AlexNet

Table 9. Smaller MobileNet Comparison to Popular Models

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
0.50 MobileNet-160	60.2%	76	1.32
Squeezenet	57.5%	1700	1.25
AlexNet	57.2%	720	60

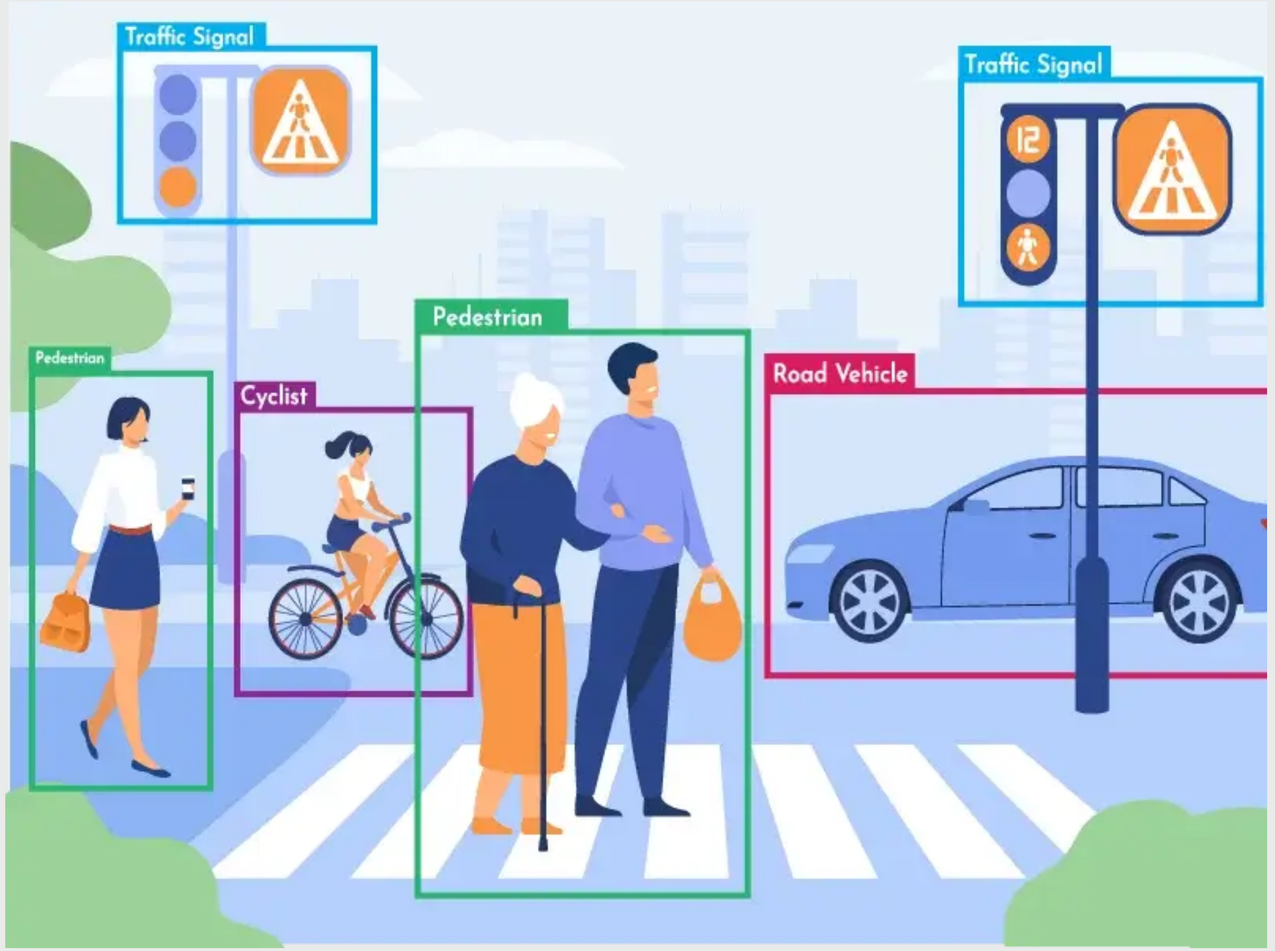
Reduced MobileNet is 4% better than AlexNet while being 45 times smaller and 9.4 times less compute.
It is also 4% better than Squeezenet at about the same size and 22 times less computation.

Object Detection

Object detection is a computer vision task in machine learning.
It involves identifying and locating objects within an image or a video.

Object detection uses applications in a variety of domains, including autonomous driving, robotics and medical imaging.

It enables machines to understand the visual world.



Experiments

COCO Object Detection

The COCO dataset contains over 200,000 images, each annotated with object labels, bounding boxes, and segmentation masks. The objects in the images are organized into 80 different categories, such as person, car, dog, chair, etc.

SSD is evaluated with 300 input resolution (SSD 300) and FasterRCNN is compared with both 300 and 600 input resolution (FasterRCNN 300, FasterRCNN 600).

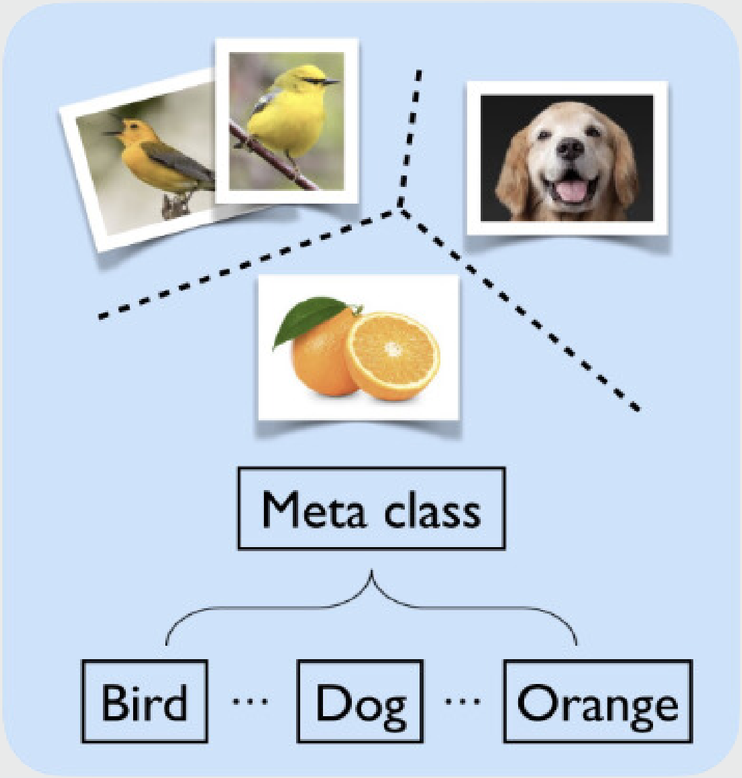
For all frameworks, MobileNet achieves comparable results to other networks with only a fraction of computational complexity and model size.

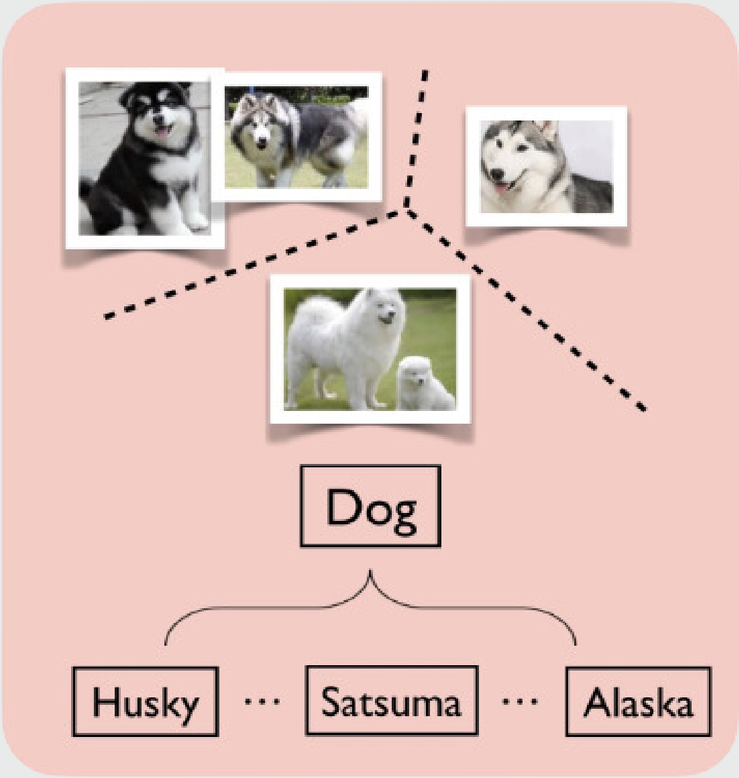
Framework Resolution	Model	mAP	Billion Mult-Adds	Million Parameters
SSD 300	deeplab-VGG	21.1%	34.9	33.1
	Inception V2	22.0%	3.8	13.7
	MobileNet	19.3%	1.2	6.8
Faster-RCNN 300	VGG	22.9%	64.3	138.5
	Inception V2	15.4%	118.2	13.3
	MobileNet	16.4%	25.2	6.1
Faster-RCNN 600	VGG	25.7%	149.6	138.5
	Inception V2	21.9%	129.6	13.3
	Mobilenet	19.8%	30.5	6.1

Fine Grained Recognition

Fine-grained recognition in machine learning refers to the task of distinguishing between highly similar subcategories within a broader category.

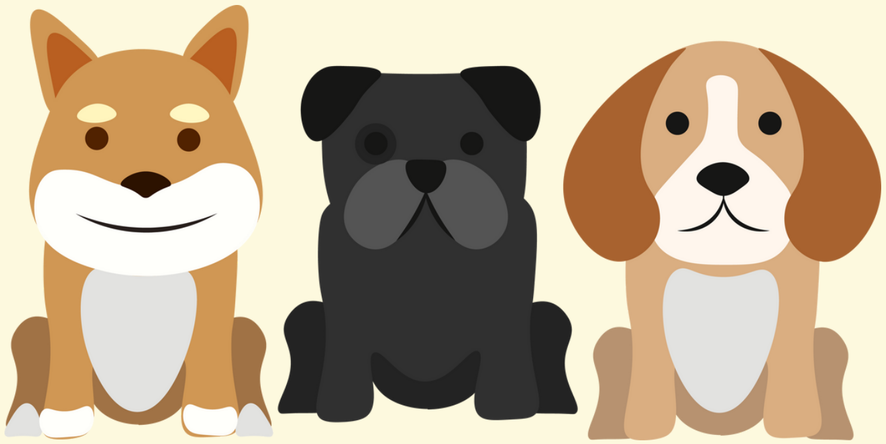
It involves training models to accurately recognize and differentiate subtle differences between objects or entities that belong to the same general class.





Experiments

Fine Grained Recognition



The training set of the Stanford Dogs dataset consists of images of 120 different dog breeds, with a varying number of images per breed.

The reaserchers pretrained a fine grained dog recognition model on noisy web data and fine tuned the model on the Stanford Dogs training set.

MobileNet can almost achieve the state of the art results from at greatly reduced computation and size.

Table 10. MobileNet for Stanford Dogs

Model	Top-1 Accuracy	Million Mult-Adds	Million Parameters
Inception V3 [18]	84%	5000	23.2
1.0 MobileNet-224	83.3%	569	3.3
0.75 MobileNet-224	81.9%	325	1.9
1.0 MobileNet-192	81.9%	418	3.3
0.75 MobileNet-192	80.5%	239	1.9

Conclusion

The researchers were able to find a model that greatly optimizes the speed and size while its accuracy is either similar or insignificantly smaller, and its computational cost is optimized.

In fact, you can say that they "just" reduced dimensions, but they managed to do it in a way that would not harm the accuracy.

Thank You!

Any Questions?