SSD Mobilenet

Object detection in embedded applications.

Sreya Francis

Overview

- SSD
- SSD vs YOLO
- Mobilenet
- SSD with Mobilenet
- Applications

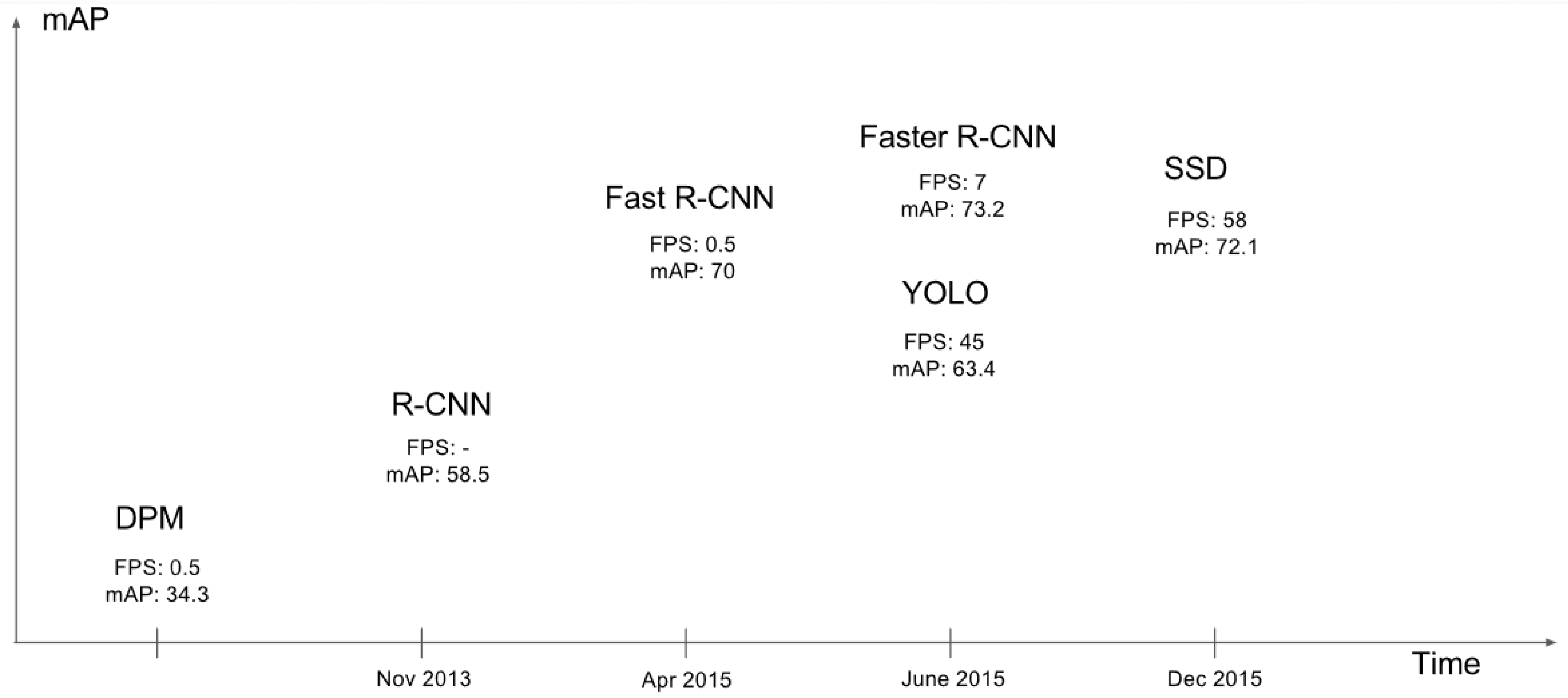
SSD

- Object detection method
- Single DNN
- Discretizes output space of bounding boxes
- Default boxes with different aspect ratios
- Scales per feature map location
- Generate scores for each object in each default box
- Combines predictions from multiple feature maps with varying resolutions.

SSD Vs YOLO

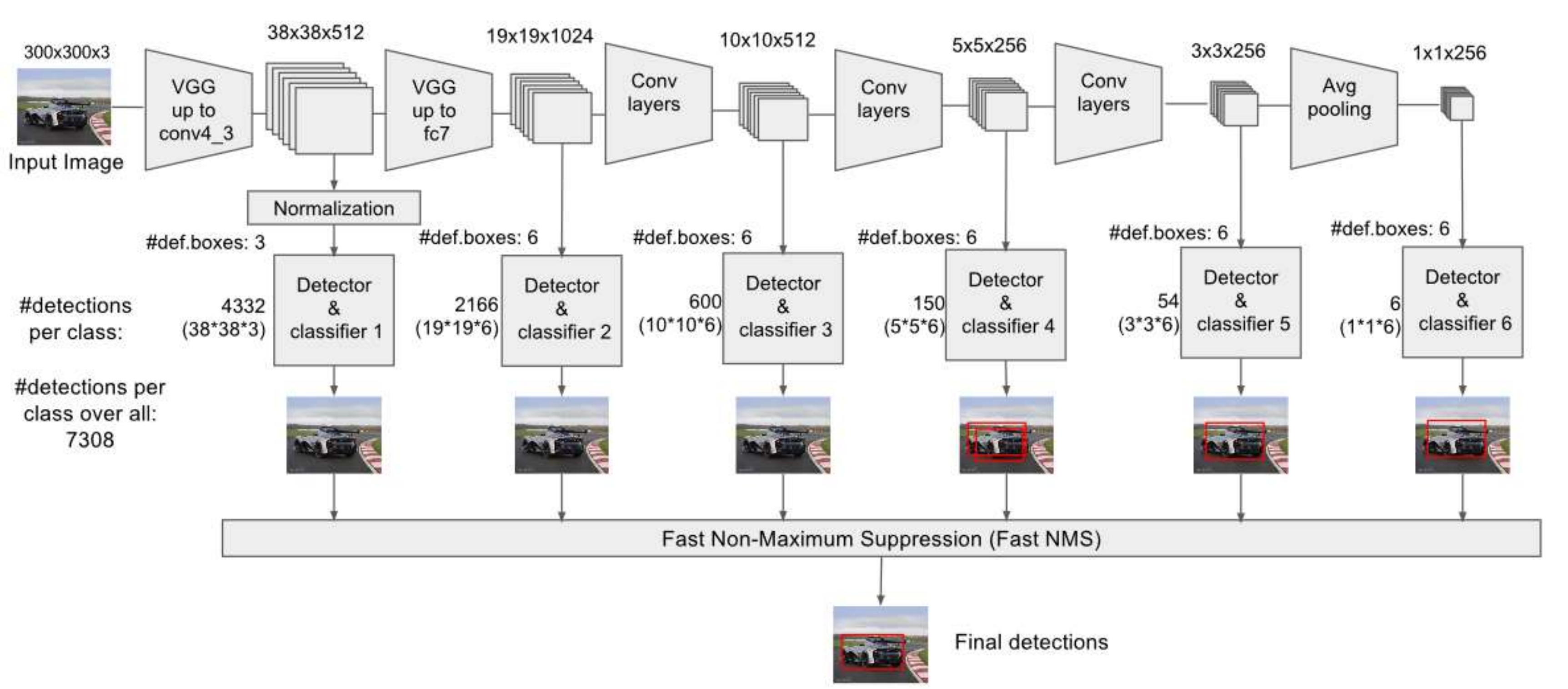
SSD	Yolo
Provides more aspect ratios. Hence wraps around object more tightly.	Predefined grid cell's aspect ratio is fixed.
Adds more convolutional layers for detection.	Uses two fully connected layers instead.
Detects objects in multiple scales better.	Multiple scale object detection is possible.

SSD Vs Yolo

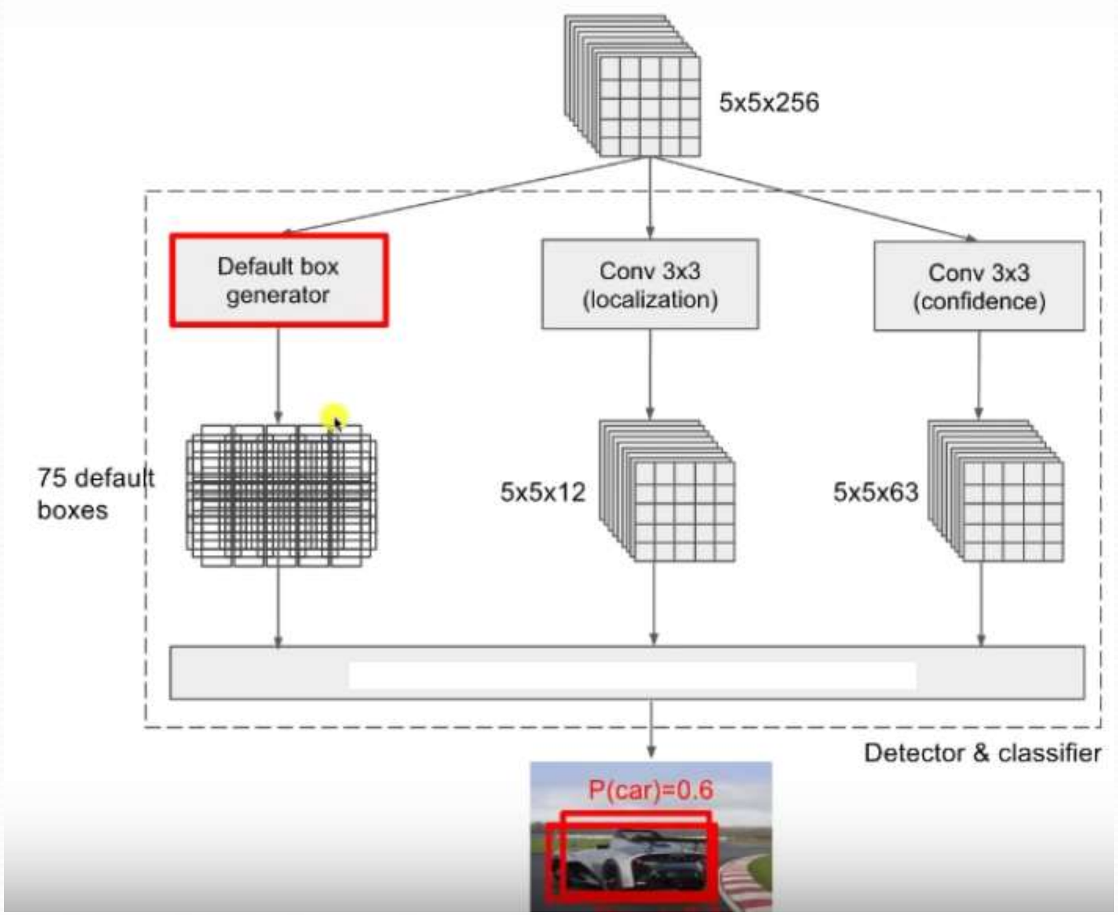


mAP
Faster R-CNN
FPS: 7
mAP: 73.2
SSD
FPS: 58
mAP: 72.1
Fast R-CNN
FPS: 0.5
mAP: 70
YOLO
FPS: 45
mAP: 63.4
R-CNN
FPS: -
mAP: 58.5
DPM
FPS: 0.5
mAP: 34.3
Nov 2013
Apr 2015
June 2015
Dec 2015
Time

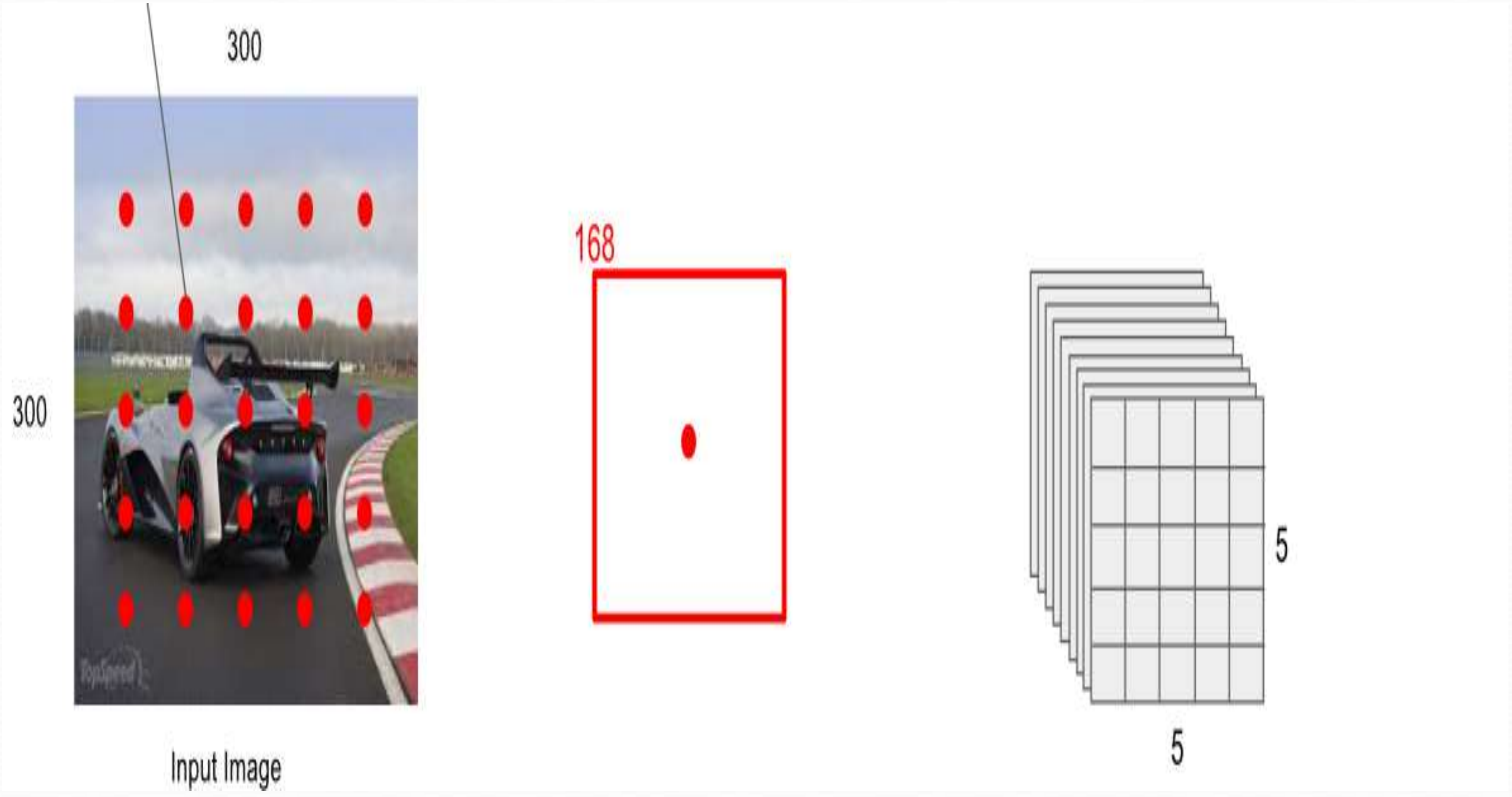
SSD Architecture

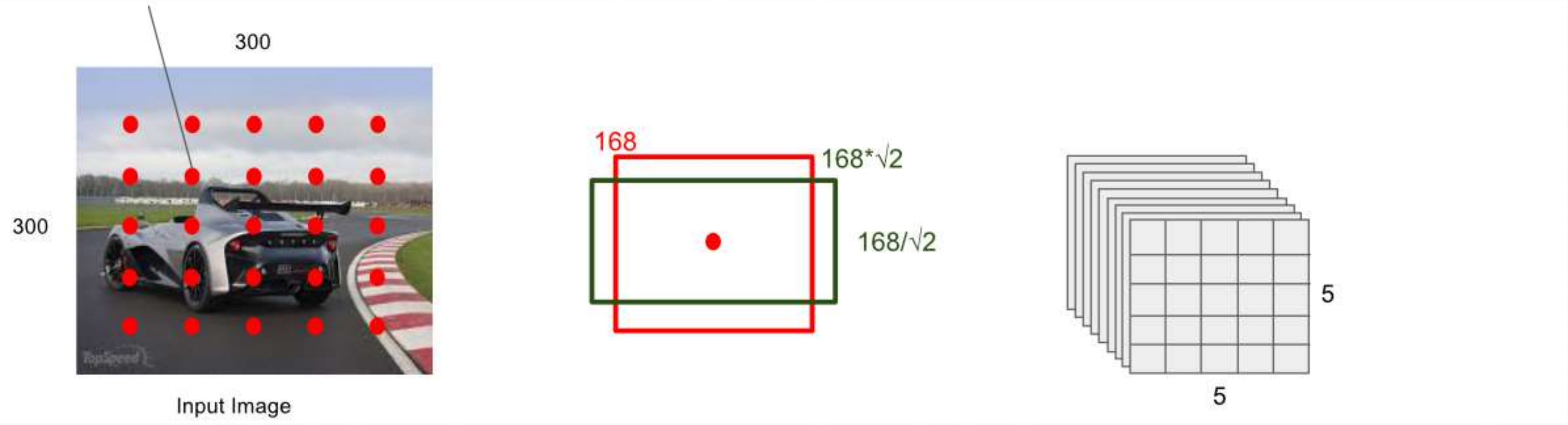


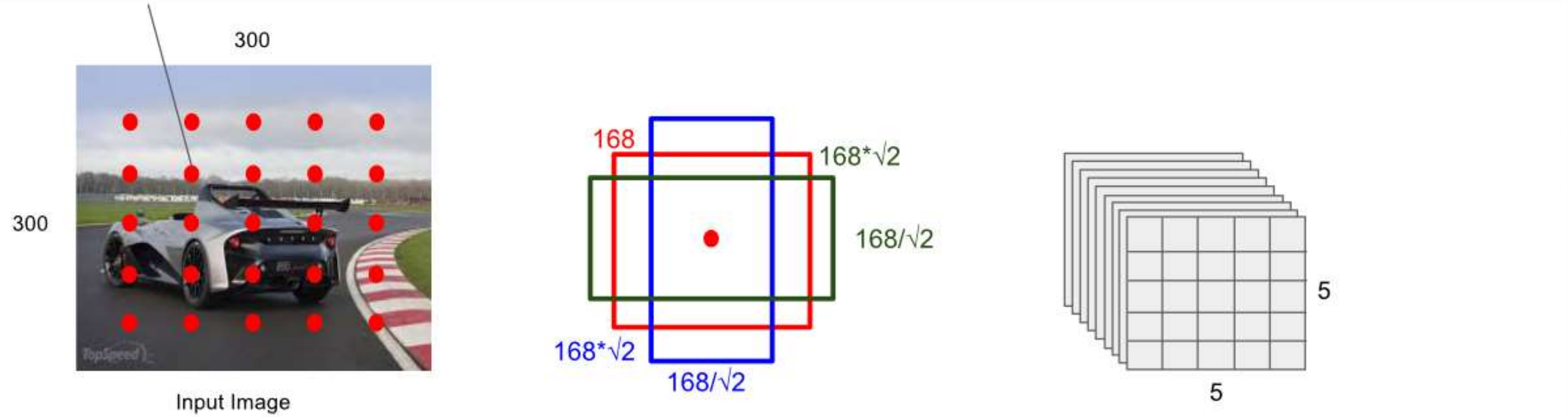
Detector + Classifier

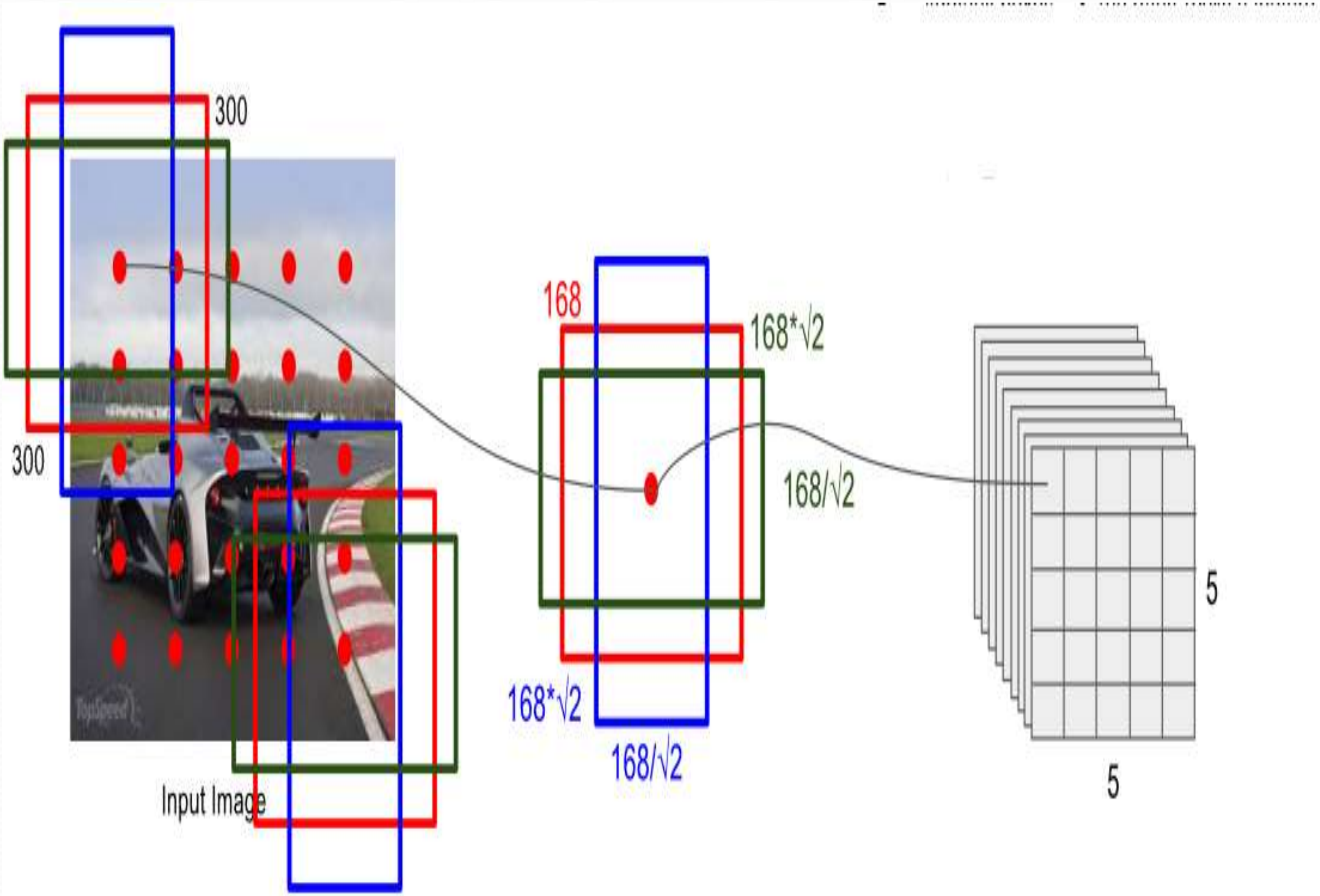


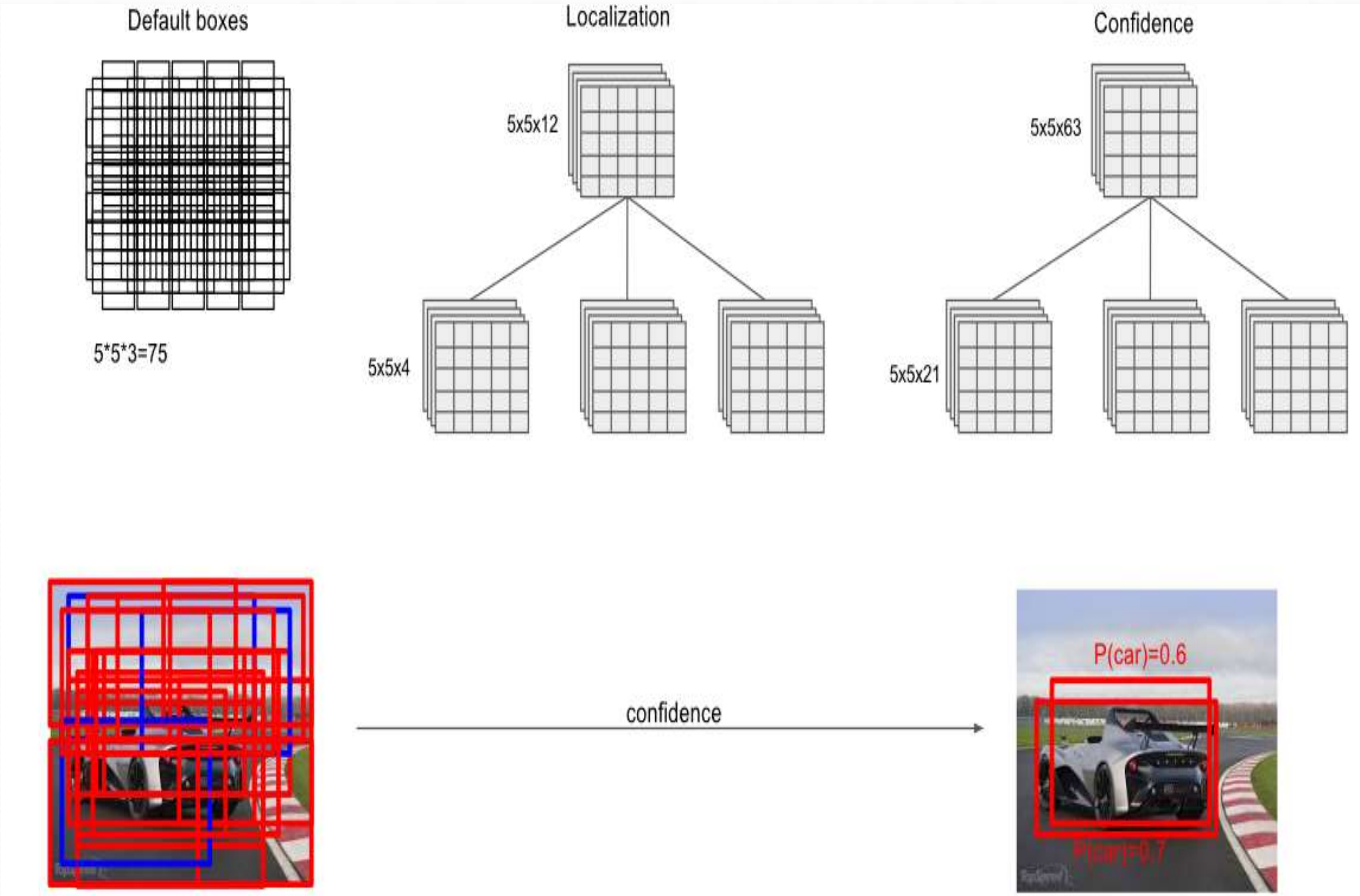
Default boxes



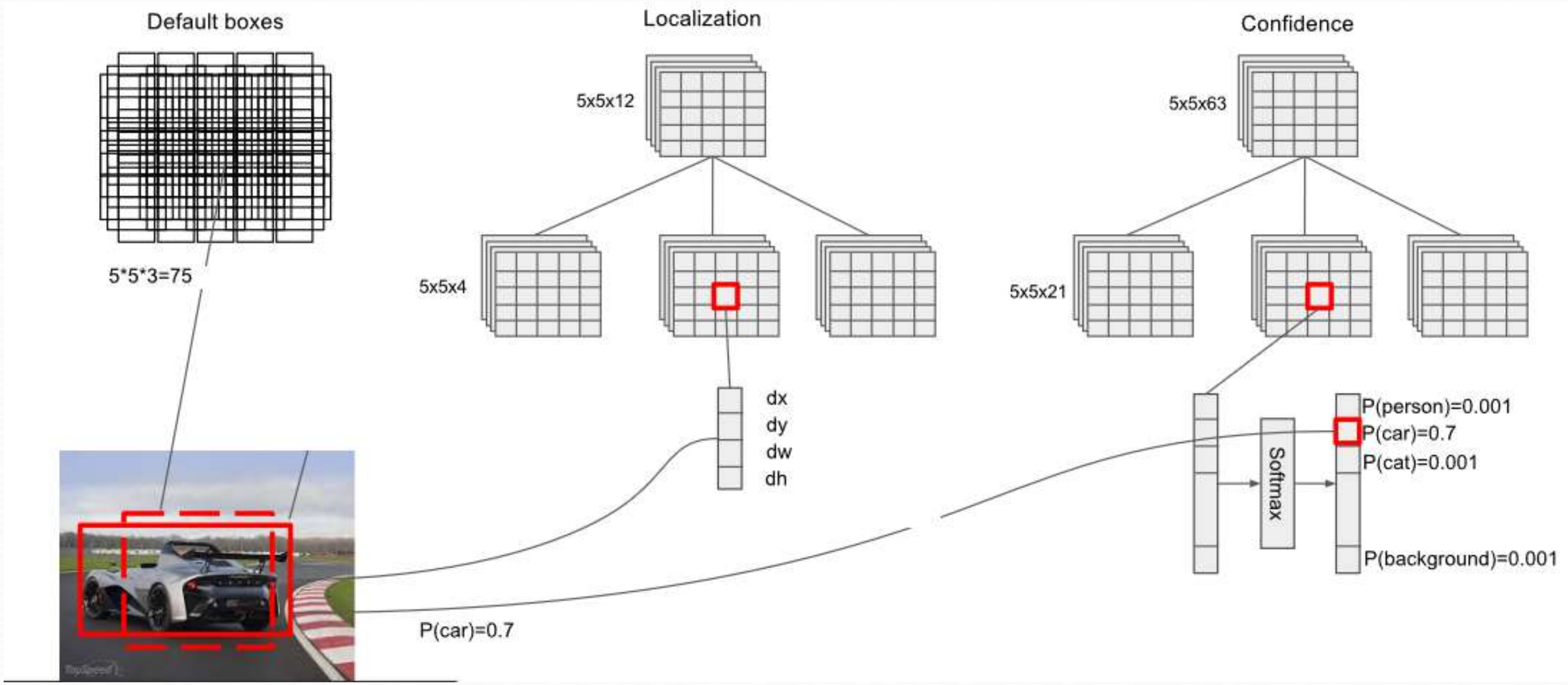

300
300
Input Image
168
168*√2
168/√2
5
5


300
300
Input Image
168
168*√2
168/√2
168*√2
168/√2
5
5


300
300
Input Image
168
168*√2
168/√2
168*√2
168/√2
5
5


Default boxes
5*5*3=75
Localization
5x5x12
5x5x4
Confidence
5x5x63
5x5x21
confidence
P(car)=0.6

SSD Detection Process



MobileNets

- model for mobile and embedded vision applications.

Definition

- Backbone Architecture – Mobile Nets
- Class of efficient models
- Proposed by Google
- Designed specifically for mobile and embedded vision applications.

Classifier

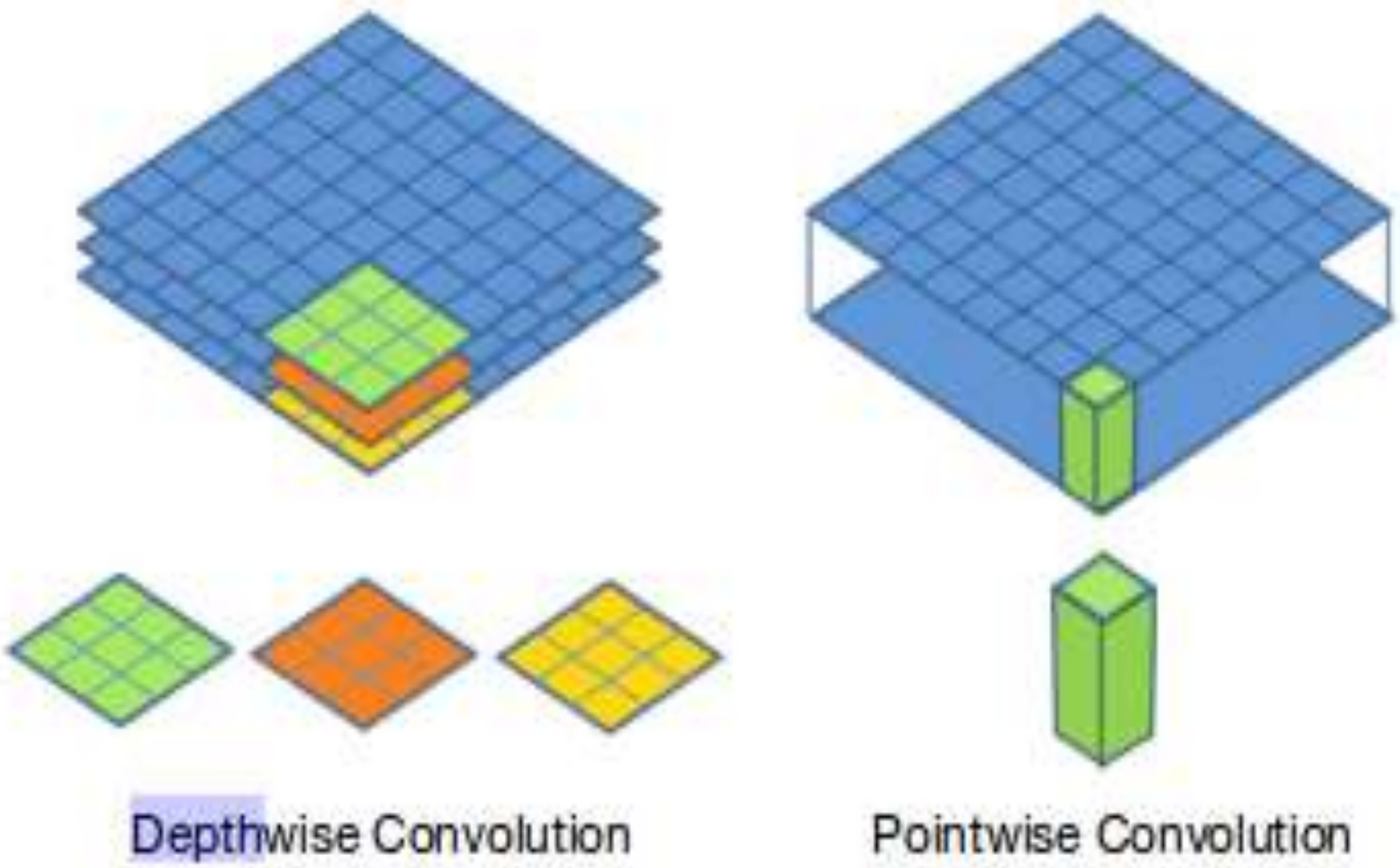
MOBILENET

- Based on a streamlined architecture
- Uses depthwise separable convolutions
- Build light weight deep neural networks

Depth wise Separable Convolution Layer

- Replaces the standard convolution with a two-step operation.
- Each Df x Df filter is only in charge of filtering a single depth of the input feature map
- Point wise convolution : A simple 1 x 1 convolution layer that is used for combining channel information.

DSC steps



Convolution & DSC

- Comparison

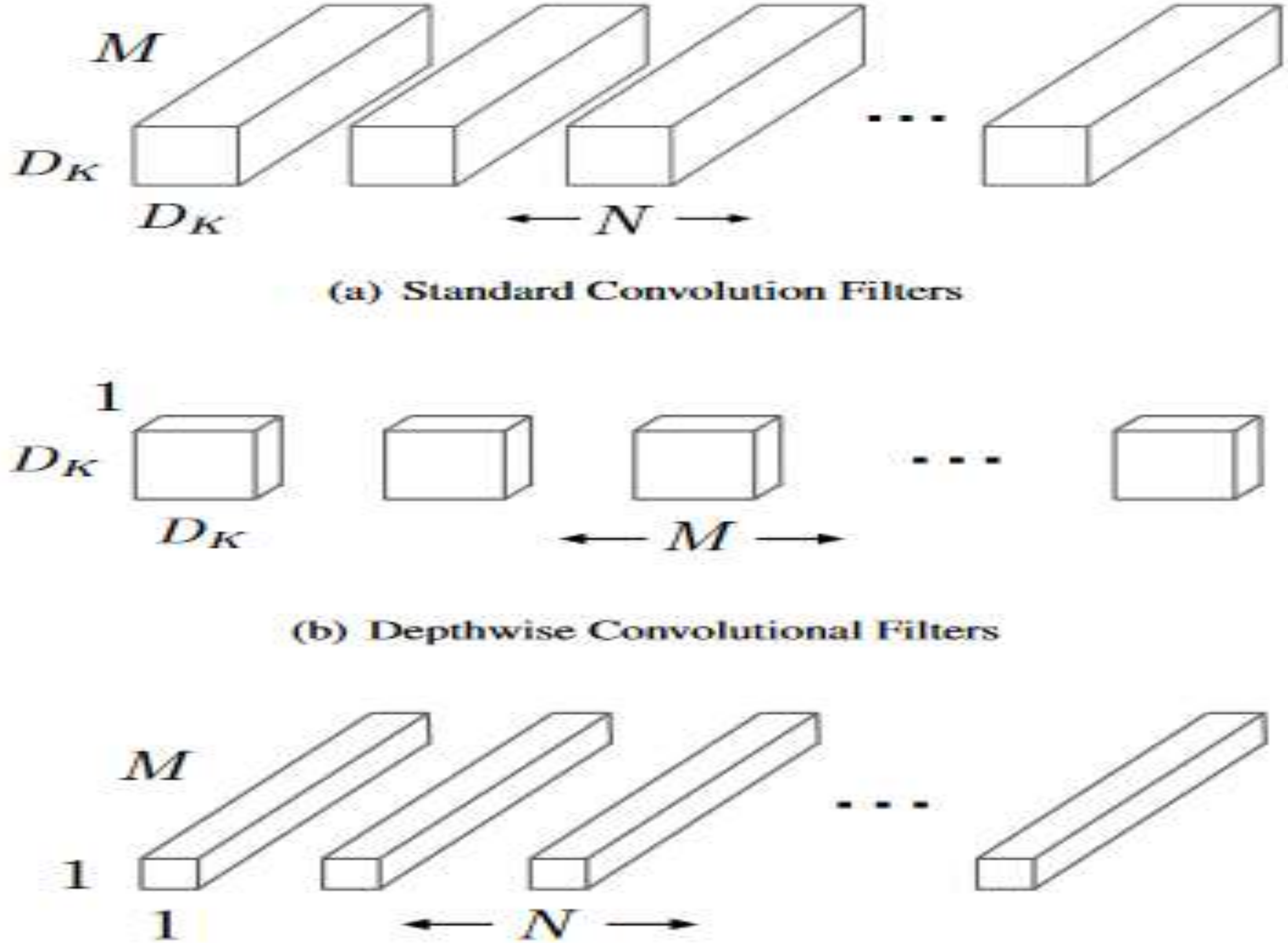
Convolution

3 x 3 conv layer	16 input channels	32 output channels
32 3x3 kernels	Each input channel is traversed.	16 x 32 feature maps
Merge one feature map By adding them up.	Out of every input channel	Done 32 times. ->>> 32 output channels
Result------------------------>>	16 x 32 x 3 x 3 parameters	------->>4608 parameters

DSC

3 x 3 conv layer	16 input channels	32 output channels
1 3x3 kernel	Each input channel is traversed.	16 feature maps
----Before Merging----		
32 1x1 convolution	Each of feature map is traversed.	32 feature maps
Result------------------------->>	16x3x3 + 16x32x1x1 parameters	------->>656 parameters

Filter Comparison



(c) 1 × 1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

Parameter and Cost

Layer	Parameter Size	Computation Cost
Standard Conv	$F \times F \times C_1 \times C_2$	$F \times F \times D_M \times D_M \times C_1 \times C_2$
Depthwise Separable	$F \times F \times C_1 + 1 \times 1 \times C_1 \times C_2$	$F \times F \times D_M \times D_M \times C_1 + 1 \times 1 \times C_1 \times C_2$

Reduction of parameters and cost

The reduction of computation cost is therefore:

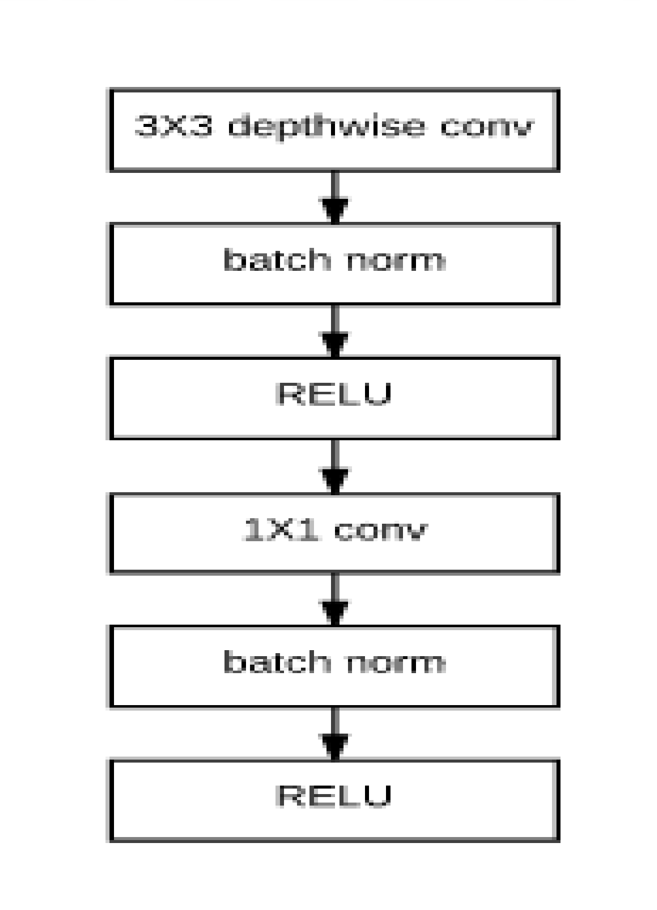
$$\frac{F \times F \times D_M \times D_M \times C_1 + C_1 \times C_2 \times D_N \times D_N}{F \times F \times D_M \times D_M \times C_1 \times C_2 \times D_N} = \frac{1}{N} + \frac{1}{F^2} \quad (1)$$

And hence the parameter reduction:

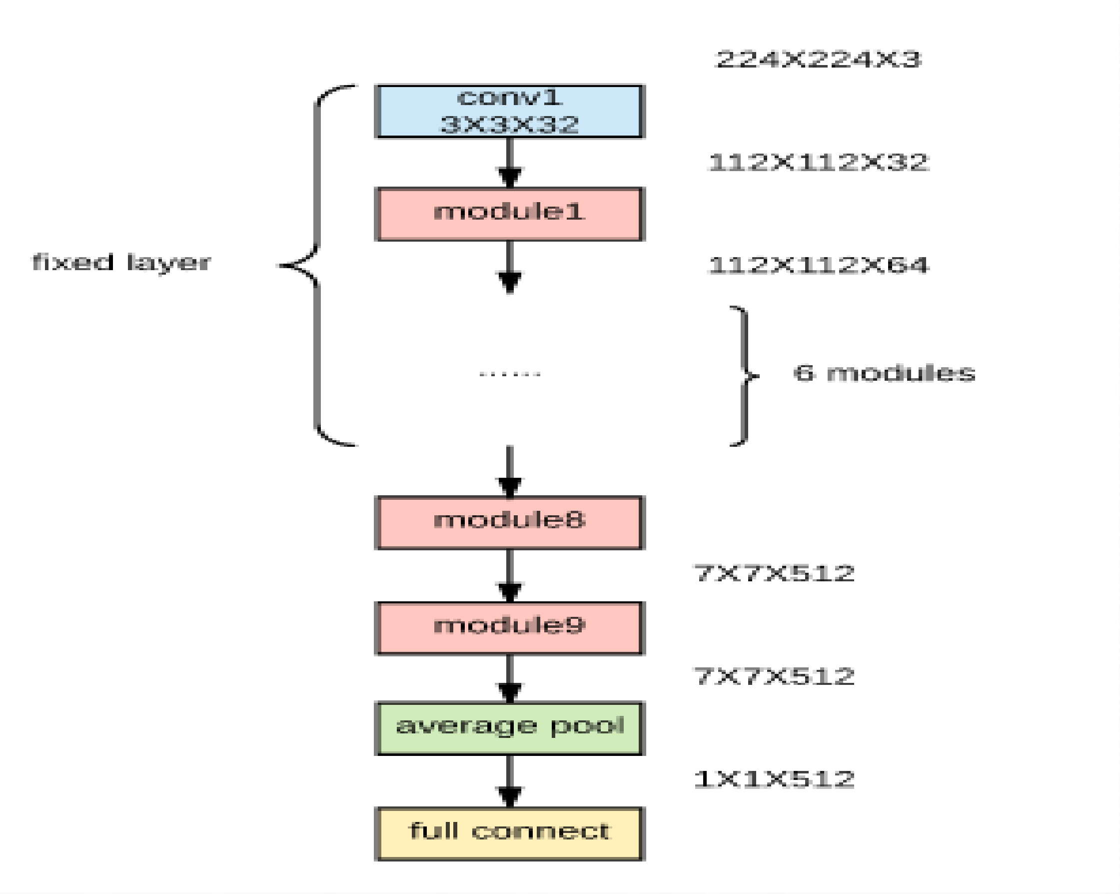
$$\frac{F \times F \times C_1 + 1 \times 1 \times C_1 \times C_2}{F \times F \times C_1 \times C_2} = \frac{1}{C_2} + \frac{1}{F^2} \quad (2)$$

Mobile net with DSC module

A DSC module



Mobilenet classifier



Mobile net classifier accuracy

cited from the Google paper [11]

Model	Dataset	Accuracy (Top-1)
MobileNets-Full	ImageNet	70.5%
VGG16	ImageNet	71.5%
MobileNets-Full	Customized VOC	69.8%
MobileNets-Small	Customized VOC	67.6%

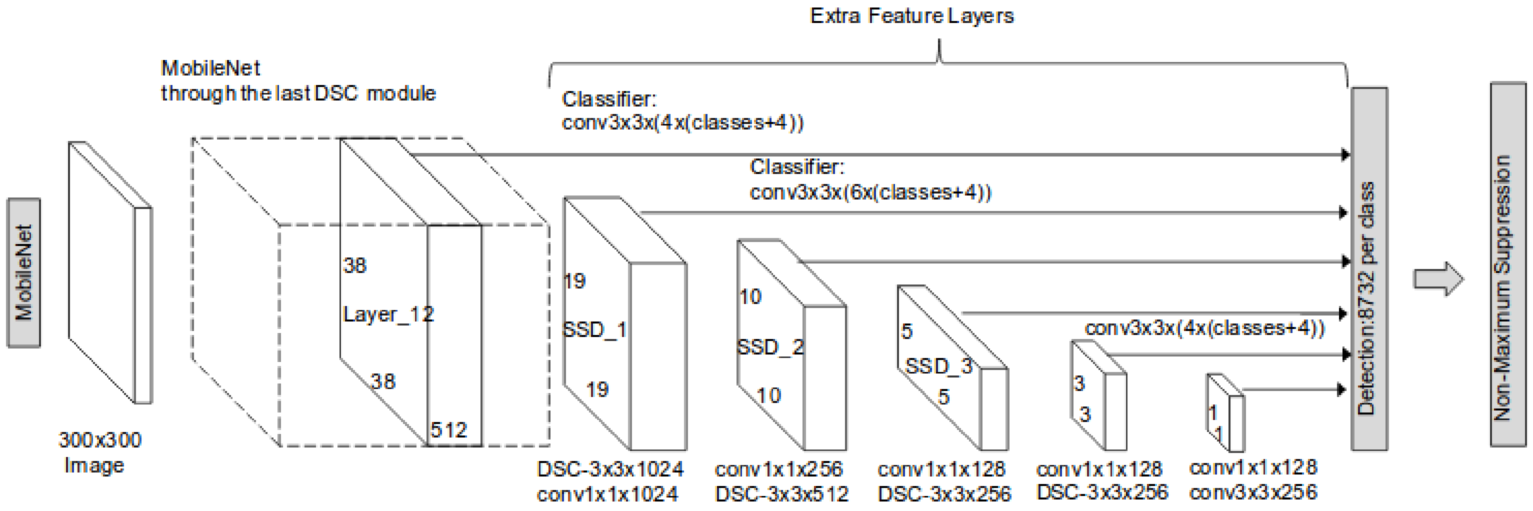
Classifier + Detector

SSD with Mobile net

SSD Mobilenet

- Mobile first object detection model
- Maximize accuracy
- Restricted resources
- Low power
- Low latency
- Small

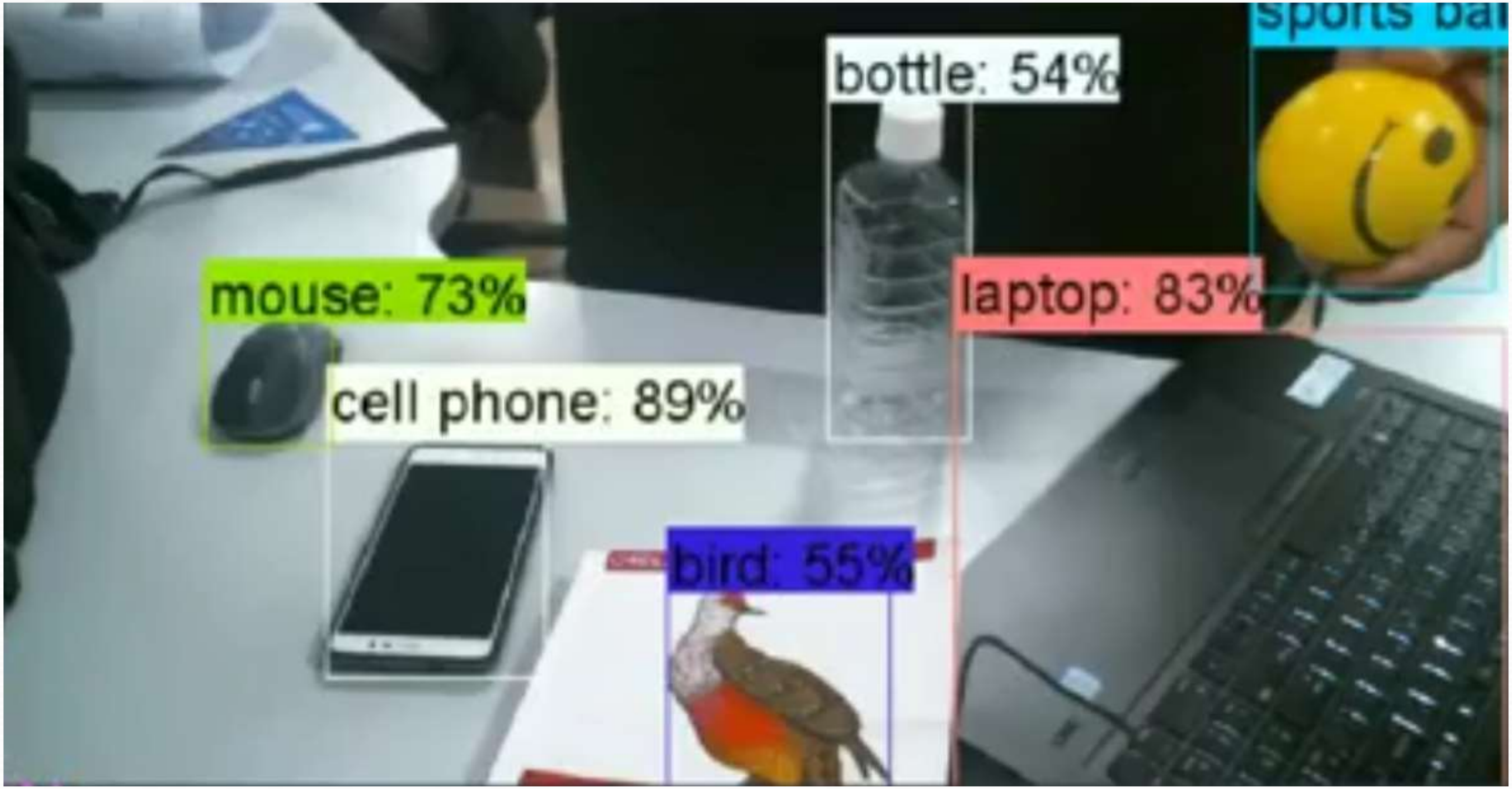
SSD Mobile net Architecture



COCO object detection results comparison using different frameworks and network architectures. mAP is reported with COCO primary challenge metric (AP at IoU=0.50:0.05:0.95)

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

SSD Mobilenet Detection Example in Tf detect android app



Tf Detect(Object Detection in Android OS)

Comparison of different models

Type of model	Size on disk	Detection speed	mAP
Yolo2-full	269.9MB	Out of memory	76.8
Yolo2-tiny	60.5MB	0.487fps	57.1
Yolo2-full eight-bit	64.4MB	0.153fps	61.3
Yolo2-tiny eight-bit	15.2MB	0.343fps	49.8
Temporal Detection	4.4MB	2.566fps	*
Mobile-Det	27.5MB	0.712fps	41.9

Thank You!