Image Processing in MobileNet CNN for Object Detection

By Josias Moukpe

What is Image Processing?

Digital image processing is the use of a digital computer to process digital images through an algorithm.

Among those algorithms we have Image Filtering, Image Padding, Image Transformation, Image Segmentation, etc.

One great application of those Image processing algorithm is Computer Vision were they are employed in Convolutional Neural Networks (CNN) to Identify Objects in Images.

Demo

Demo End

COCO SSD MobileNet v1 CNN

MobileNet is a light weight (27Mb) Convolutional Neural Network (CNN) used for Classification of object. Unleashed on a video stream, it can identify objects and even infer a bounding box of the object

MobileNet is provided as Tensorflow Lite model by Google. It was trained on the Common Object in Context (COCO) dataset.

It is a Single Shot Detector (SSD), meaning it only needs 1 feedforward pass of the input through its layers for inference.

Architecture

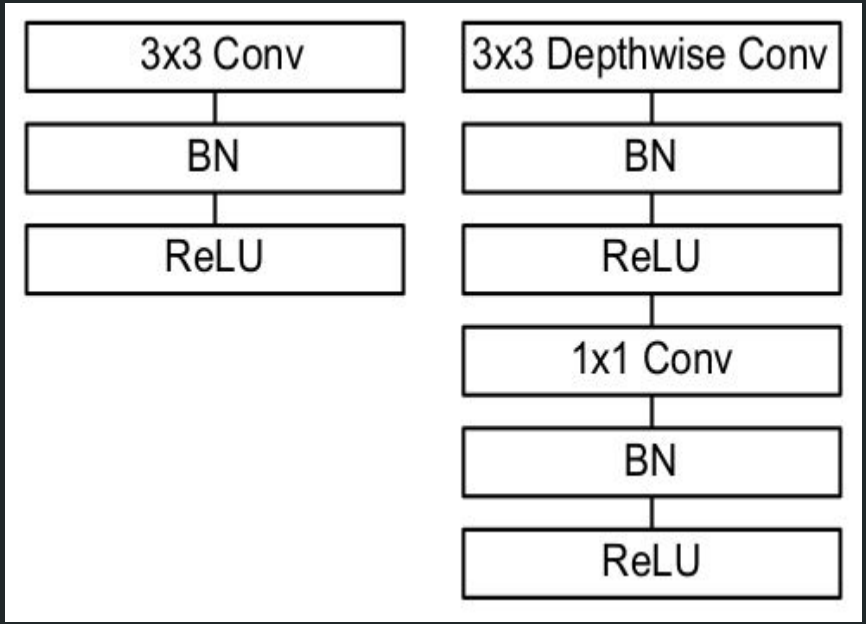


Table 1. MobileNet Body Architecture

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	$1 \times 1 \times 1000$

Image Processing Involved (Continued)

Outside the Neural Network

The image is captured from the camera

The image is converted to RGB

The image is then resized to fit the input of the MobileNet (1x224x224x3) RGB image

Inside the Neural Network

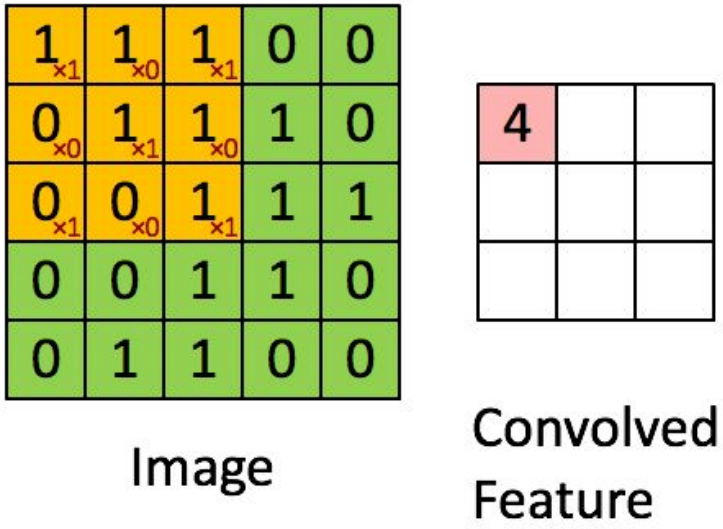
At Every layer of the Network, the image either goes through a Normal Convolution layer or a Depthwise Convolution layer.

Every convolution layer extract different features such as edges, colors, shapes, gradient orientations, from the image by applying filters of different values, sizes and strides.

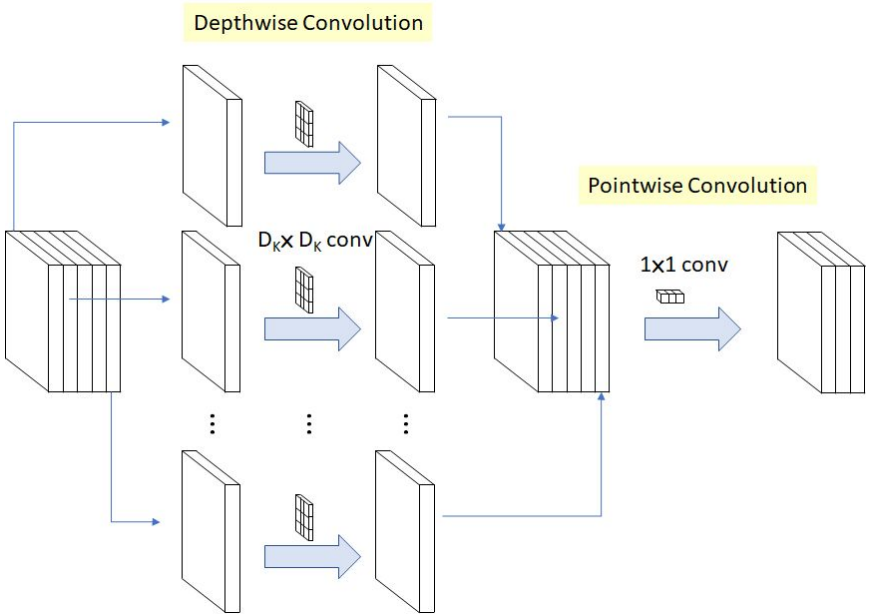
The Depthwise Convolution Layer is the crux of MobileNet, allowing it to perform detection at the edge (RPI and Mobile) despite resource constraints

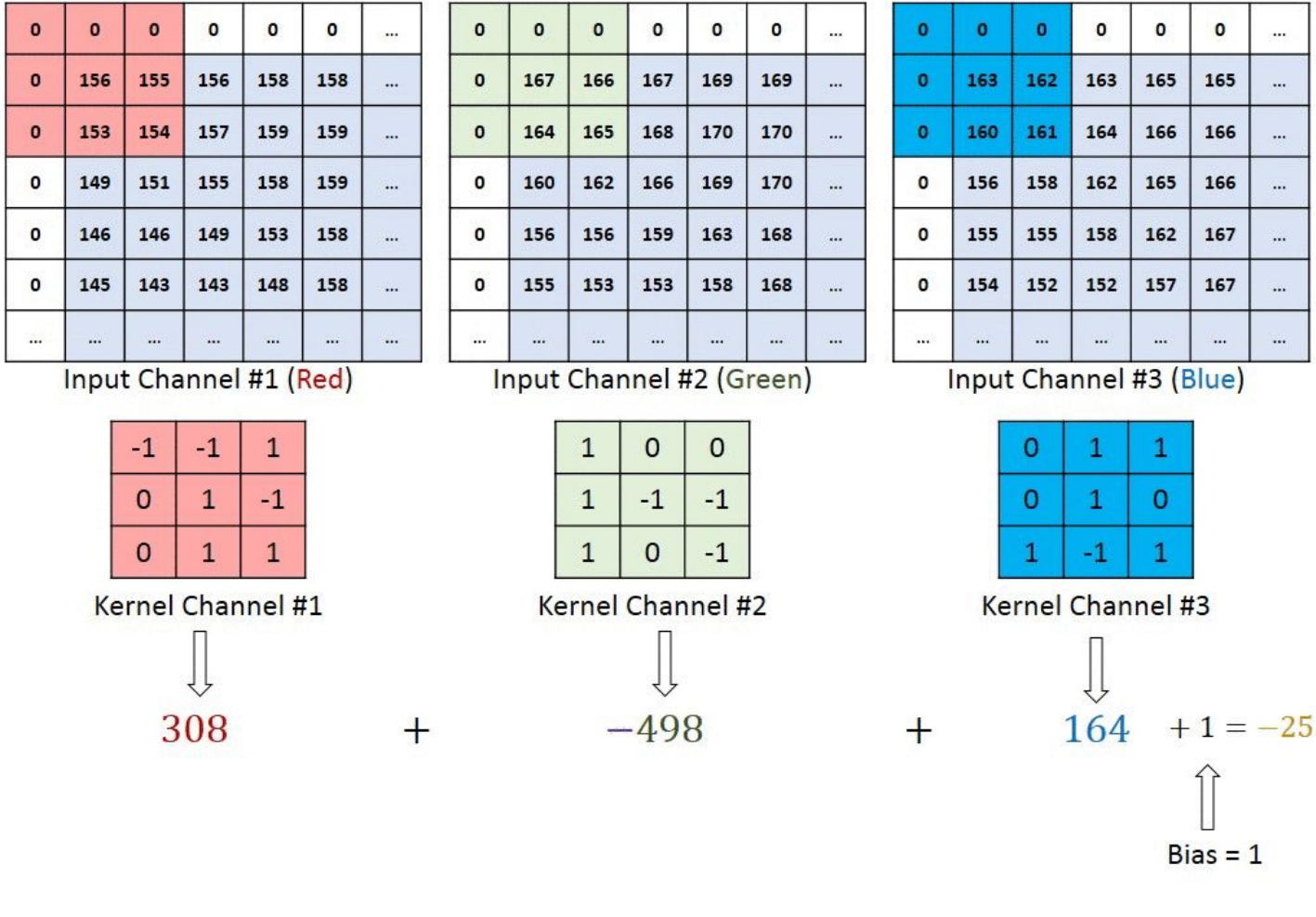
Image Processing Involved

Convolution with Filter for Attribute Detection



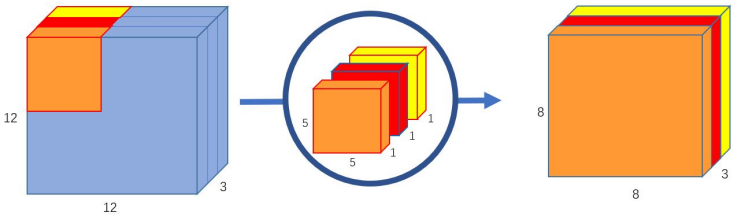
Depthwise Separable Convolution




0 0 0 0 0 0 ...
0 156 155 156 158 158 ...
0 153 154 157 159 159 ...
0 149 151 155 158 159 ...
0 146 146 149 153 158 ...
0 145 143 143 148 158 ...
... ...
Input Channel #1 (Red)
0 0 0 0 0 0 ...
0 167 166 167 169 169 ...
0 164 165 168 170 170 ...
0 160 162 166 169 170 ...
0 156 156 159 163 168 ...
0 155 153 153 158 168 ...
... ...
Input Channel #2 (Green)
0 0 0 0 0 0 ...
0 163 162 163 165 165 ...
0 160 161 164 166 166 ...
0 156 158 162 165 166 ...
0 155 155 158 162 167 ...
0 154 152 152 157 167 ...
... ...
Input Channel #3 (Blue)
-1 -1 1
0 1 -1
0 1 1
Kernel Channel #1
1 0 0
1 -1 -1
1 0 -1
Kernel Channel #2
0 1 1
0 1 0
1 -1 1
Kernel Channel #3
308
+
−498
+
164
+ 1 = −25
Bias = 1
Output
-25 ... ...
... ...
... ...
... ...
... ...

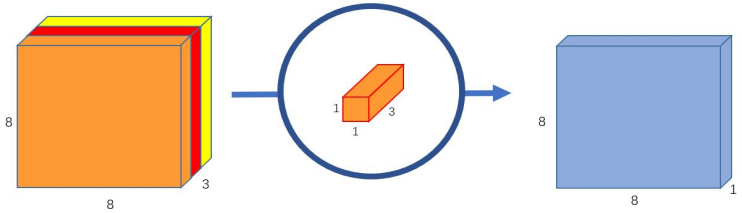
Depthwise and Pointwise convolution

Depthwise



Each 5x5x1 kernel iterates 1 channel of the image (note: 1 channel, not all channels), getting the scalar products of every 25 pixel group, giving out a 8x8x1 image. Stacking these images together creates a 8x8x3 image.

Pointwise



it uses a 1x1 kernel, or a kernel that iterates through every single point. This kernel has a depth of however many channels the input image has (here 3). Therefore, we iterate a 1x1x3 kernel through our 8x8x3 image, to get a 8x8x1 image. We can create 256 1x1x3 kernels that output a 8x8x1 image each to get a final image of shape 8x8x256.

71%

Accuracy while remaining lightweight for mobile devices and RPi

Conclusion

Image processing is used heavily in Computer Vision. With the help of Machine Learning algorithms, we are witnessing a surge in research and applications of Image Processing to all sorts of field such Medicine, Automotive, and Consumer Electronics.

A lot of the concepts seen in class have been applied to the realization of this project and contributed to a deeper understanding of Convolutional Neural Networks.

Thanks!

Questions?