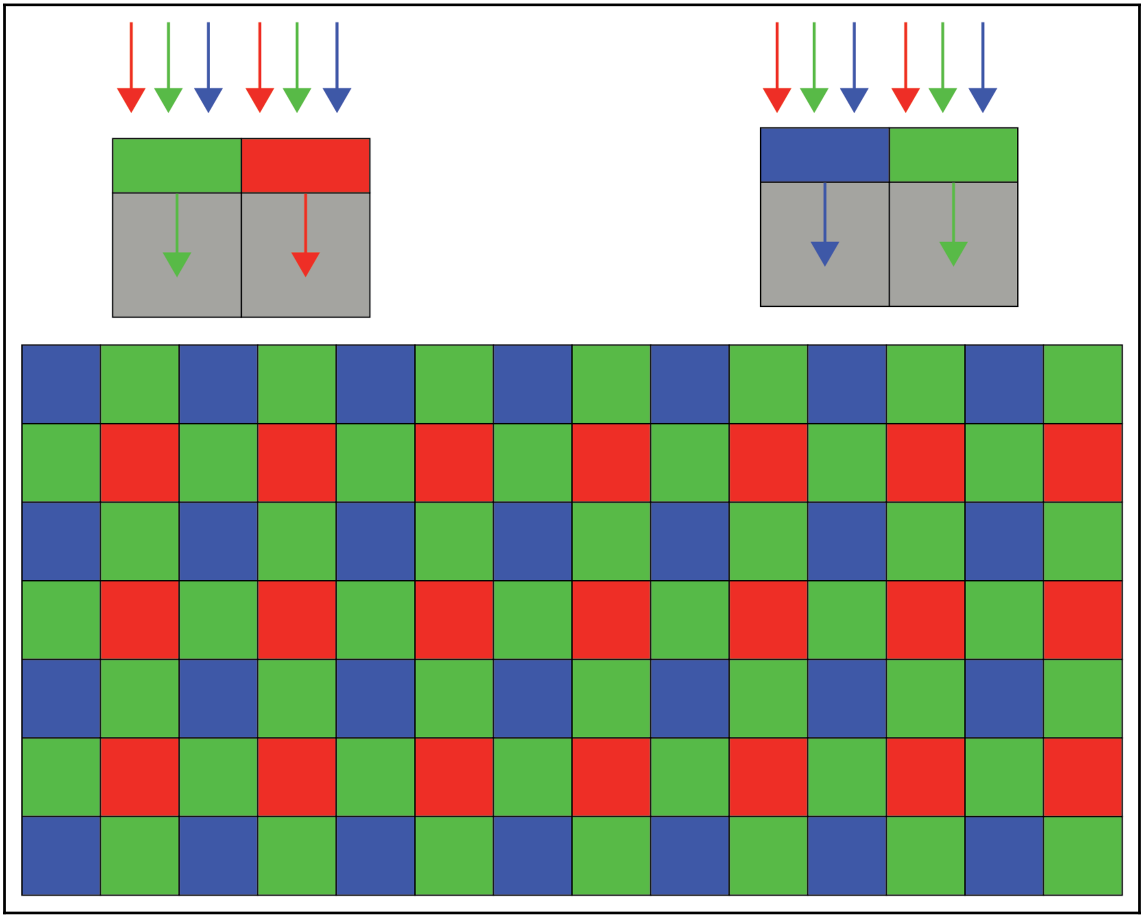
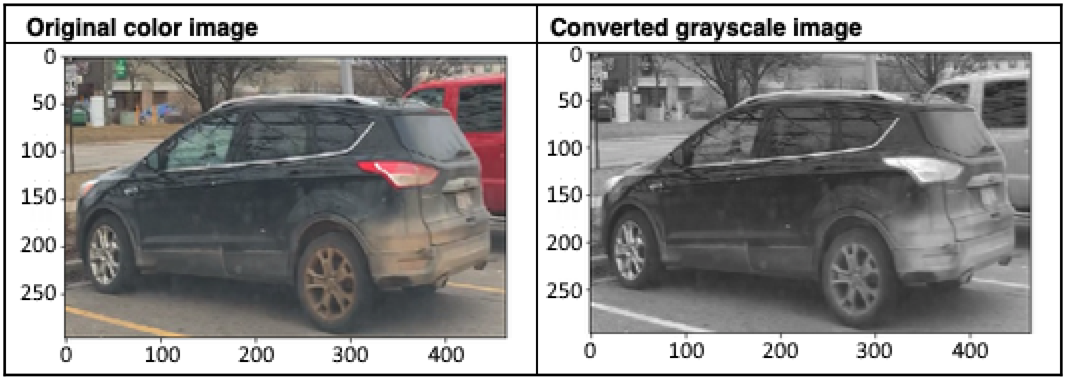
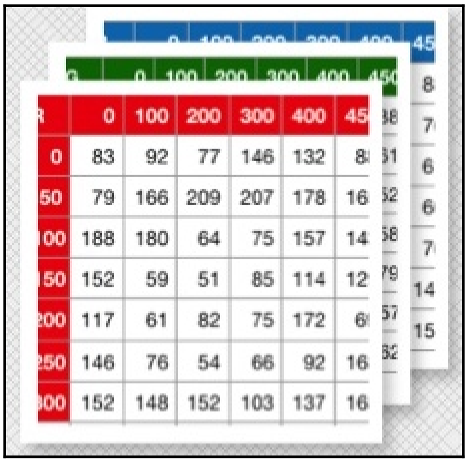
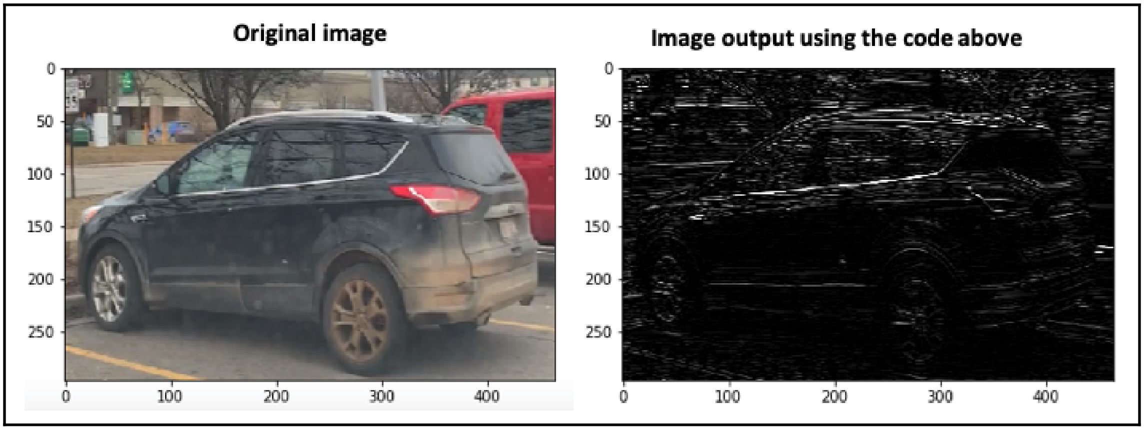
Chapter 1: Computer Vision and TensorFlow Fundamentals

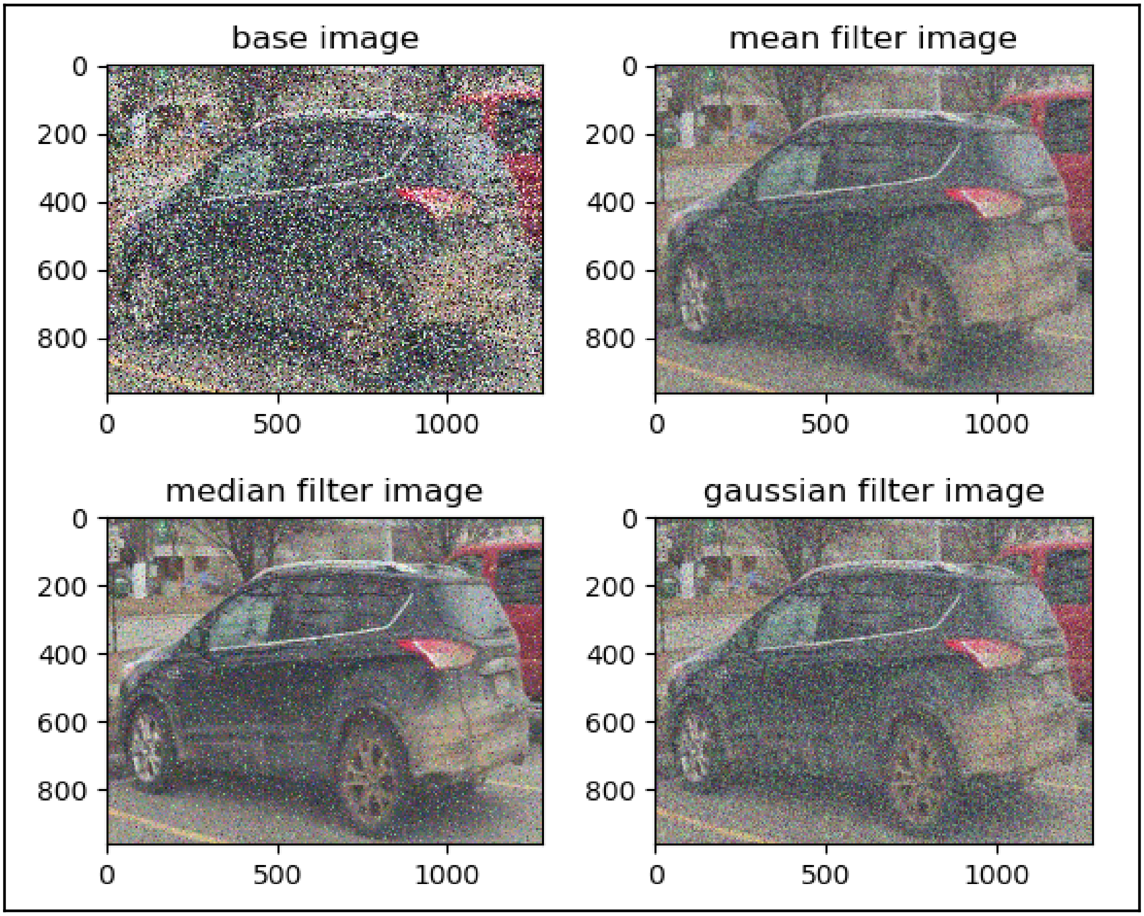






R	0	100	200	300	400	45
0	83	92	77	146	132	8
50	79	166	209	207	178	16
00	188	180	64	75	157	14
50	152	59	51	85	114	12
00	117	61	82	75	172	6
50	146	76	54	66	92	16
00	152	148	152	103	137	16


Original image
0
50
100
150
200
250
0
100
200
300
400
Image output using the code above
0
50
100
150
200
250
0
100
200
300
400


base image
0
200
400
600
800
0
500
1000
mean filter image
0
200
400
600
800
0
500
1000
median filter image
0
200
400
600
800
0
500
1000
gaussian filter image
0
200
400
600
800
0
500
1000

INPUT IMAGE PIXELS			KERNEL				CONVOLUTION			OUTPUT PIXEL	Final Image

Original image

Mean Kernel - difference between adjacent pixels blurred

165	150	138	1	1	2	1	10.313	18.8	8.625	
153	137	113	—	2	4	2	19.125	34.3	14.13	134.688
145	112	103	16	1	2	1	9.0625	14	6.438	

Gaussian Kernel - difference between adjacent pixels blurred

165	150	138	1	0.37	0.61	0.37	9.7164	14.6	8.126	
153	137	113	—	0.61	1.00	0.61	14.854	21.8	10.97	105.512
145	112	103	2PI	0.37	0.61	0.37	8.5387	10.9	6.065	

Shapen Kernel (Laplacian)- difference between adjacent pixels emphasized

165	150	138	1	0	-1	0	0	-150	0	
153	137	113	—	-1	4	-1	-153	548	-113	20
145	112	103	1	0	-1	0	0	-112	0	

INPUT IMAGE PIXELS			KERNEL				CONVOLUTION			OUTPUT PIXEL	Final Image

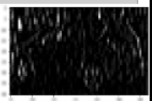
Original image

Horizontal kernel

165	150	138	1	-1	-1	-1	-165	-150	-138	
153	137	113	—	2	2	2	306	274	226	-7
145	112	103	1	-1	-1	-1	-145	-112	-103	

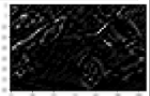
Vertical kernel

165	150	138	1	-1	2	-1	-165	300	-138	
153	137	113	—	-1	2	-1	-153	274	-113	-19
145	112	103	1	-1	2	-1	-145	224	-103	



Oblique kernel +45 degree

165	150	138	1	-1	-1	2	-165	-150	276	
153	137	113	—	-1	2	-1	-153	274	-113	44
145	112	103	1	2	-1	-1	290	-112	-103	



Shapen Kernel (Laplacian)- difference between adjacent pixels emphasized

165	150	138	1	0	1	0	0	150	0	
153	137	113	—	1	-4	1	153	-548	113	-20
145	112	103	1	0	1	0	0	112	0	

Horizontal kernel

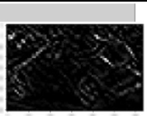
165	150	138	1	-1	-1	-1	-165	-150	-138	
153	137	113	—	2	2	2	306	274	226	-7
145	112	103	1	-1	-1	-1	-145	-112	-103	

Vertical kernel

165	150	138	1	-1	2	-1	-165	300	-138	
153	137	113	—	-1	2	-1	-153	274	-113	-19
145	112	103	1	-1	2	-1	-145	224	-103	

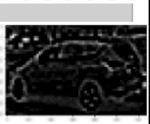
Oblique kernel -45 degree

165	150	138	1	2	-1	-1	330	-150	-138	
153	137	113	—	-1	2	-1	-153	274	-113	-1
145	112	103	1	-1	-1	2	-145	-112	206	



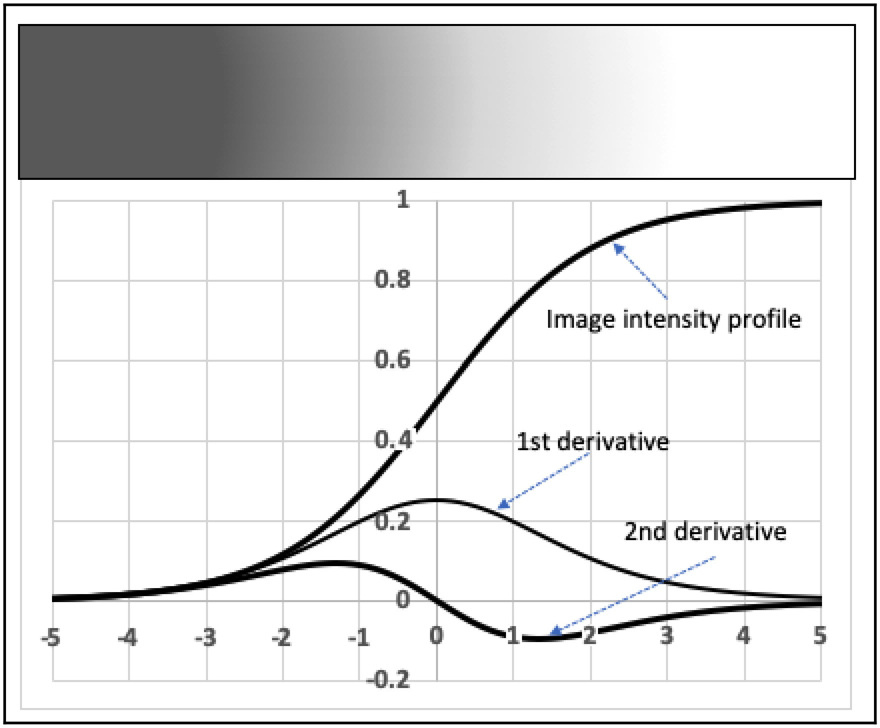
Gaussian + Laplacian

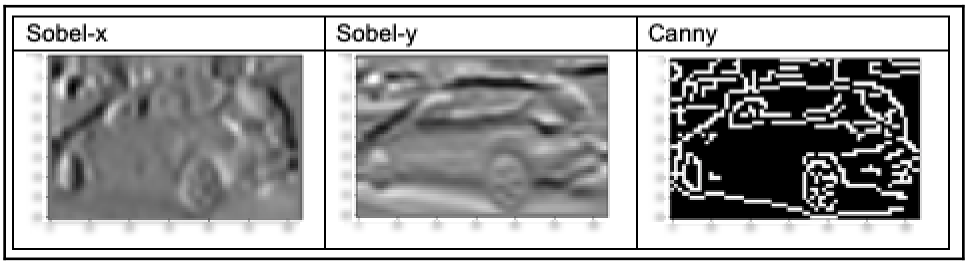
10	19	8.6	1	0	1	0	0	18.75	0	
19	34	14	—	1	-4	1	19.125	-137	14.125	-71
9.1	14	6.4	1	0	1	0	0	14	0	

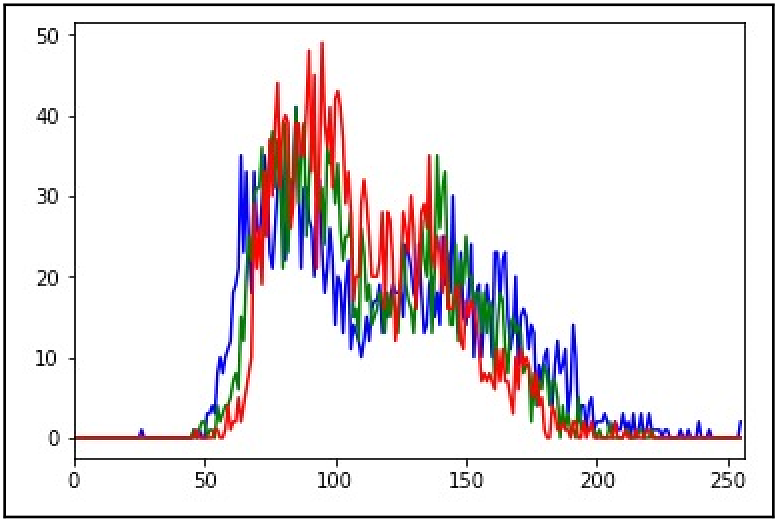


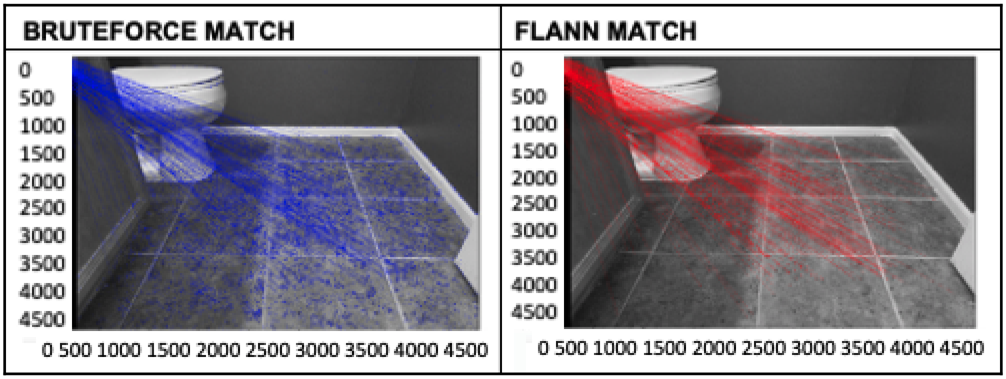
Three timess Gaussian + Laplacian

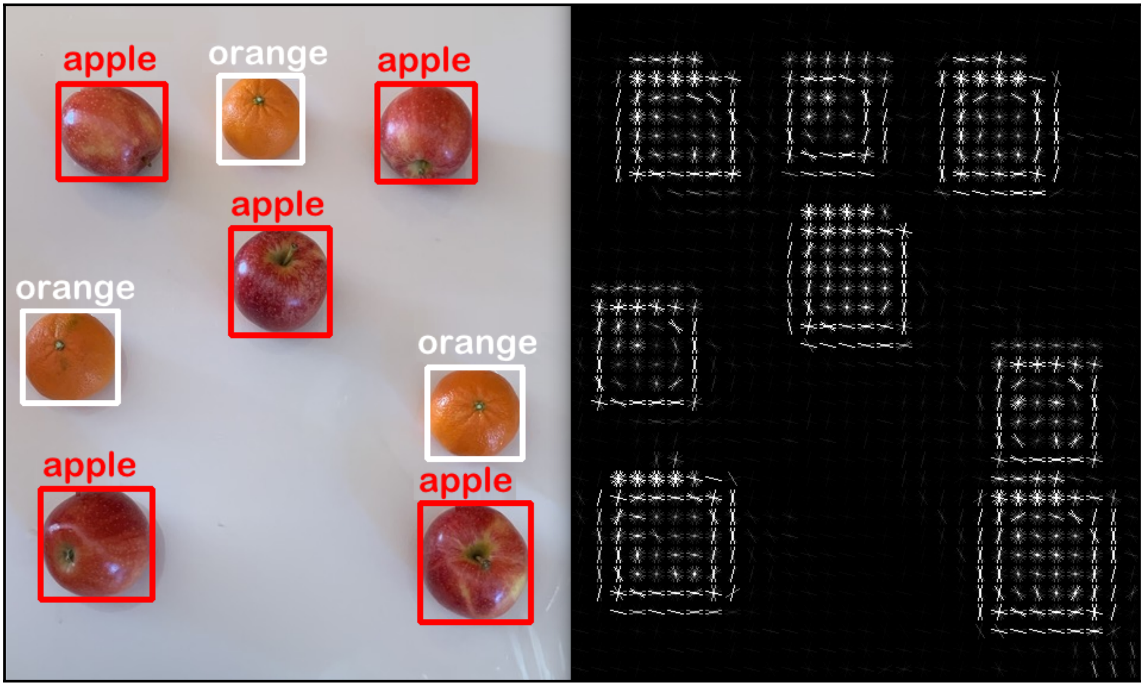
0	0.3	0	1	0	1	0	0	0.293	0	
0.3	2.1	0.2	—	1	-4	1	0.2988	-8.563	0.2207	-7.5313
0	0.2	0	1	0	1	0	0	0.2188	0	


1
0.8
0.6
0.4
0.2
0
-0.2
-5
-4
-3
-2
-1
0
1
2
3
4
5
Image intensity profile
1st derivative
2nd derivative

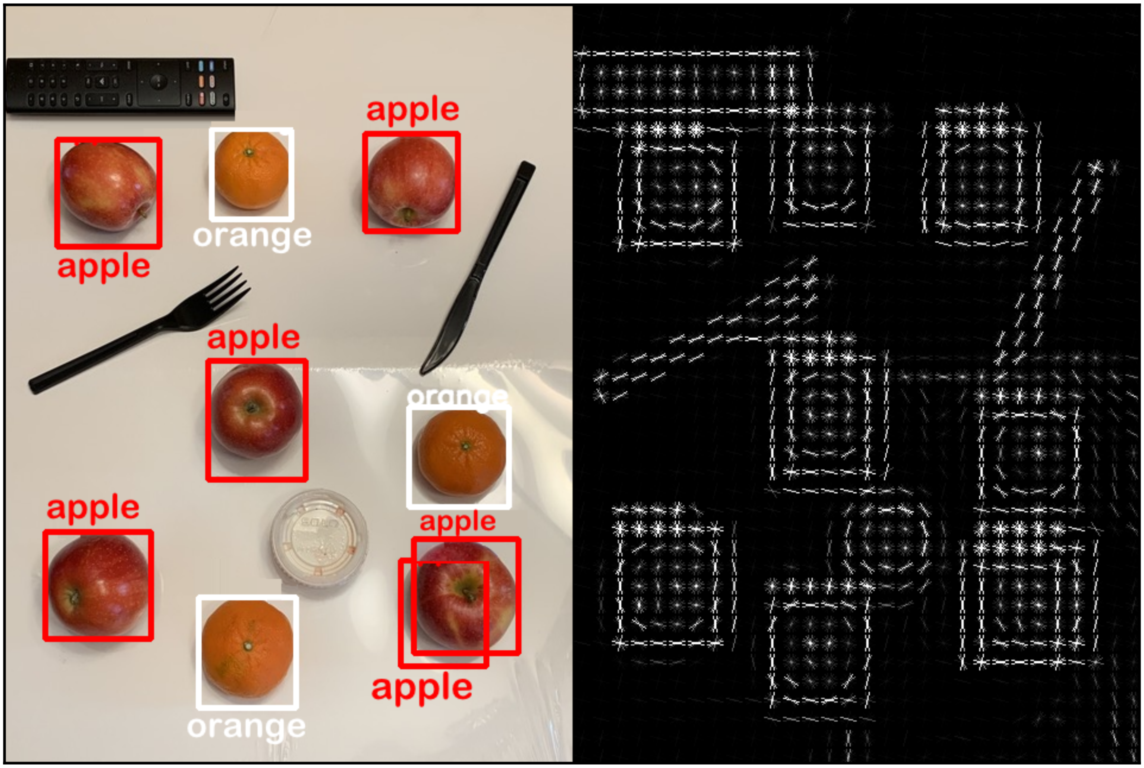

Sobel-x
Sobel-y
Canny


0
10
20
30
40
50
0
50
100
150
200
250

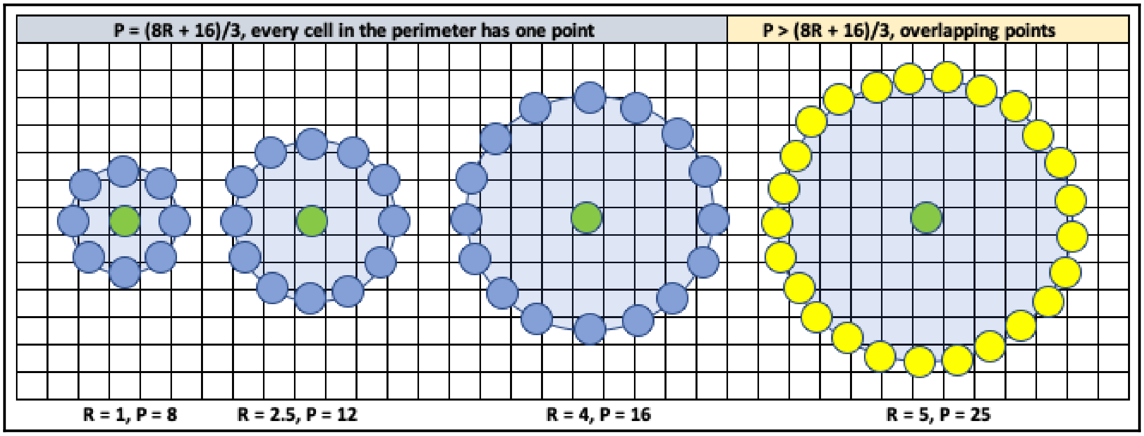

BRUTEFORCE MATCH
FLANN MATCH
0 500 1000 1500 2000 2500 3000 3500 4000 4500
0 500 1000 1500 2000 2500 3000 3500 4000 4500
0 500 1000 1500 2000 2500 3000 3500 4000 4500
0 500 1000 1500 2000 2500 3000 3500 4000 4500

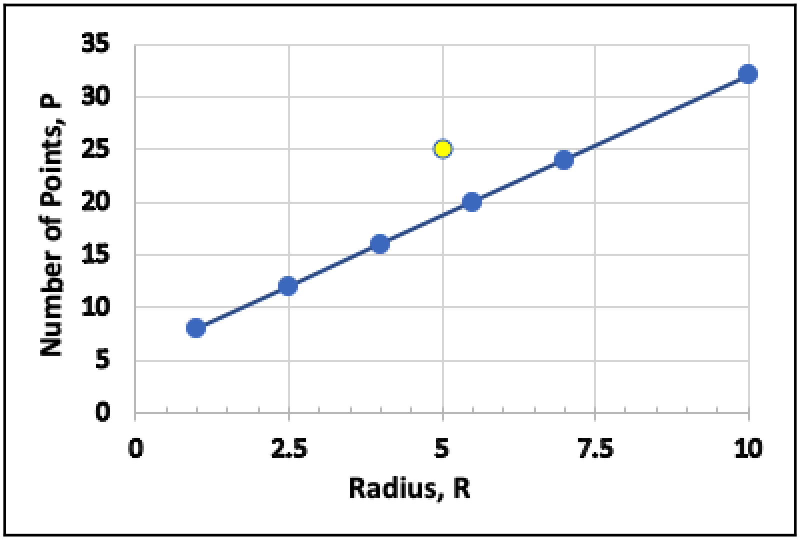


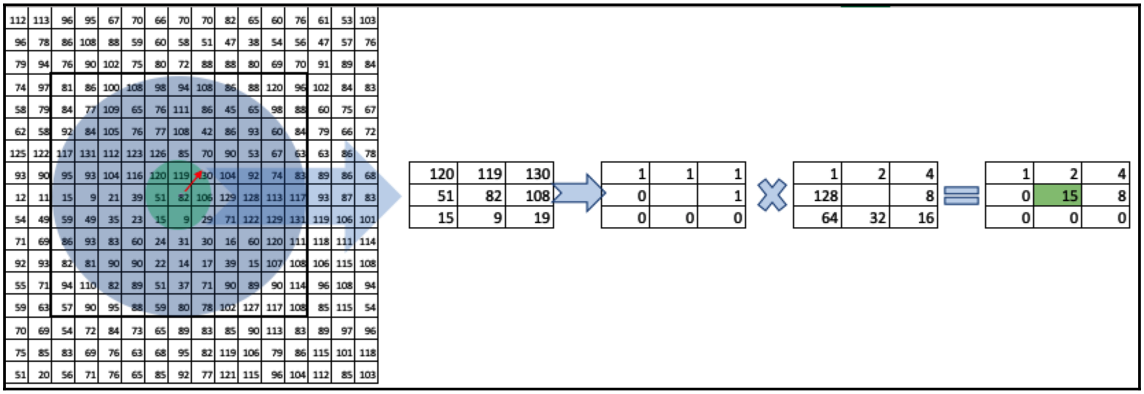
apple
orange
apple
apple
orange
orange
apple
apple


apple
orange
apple
apple
apple
orange
apple
apple
orange
apple

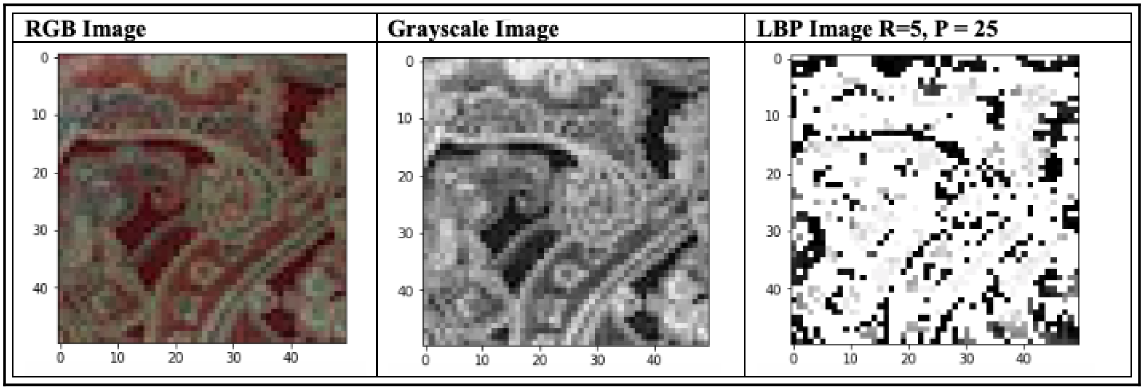
Chapter 2: Content Recognition Using Local Binary Patterns



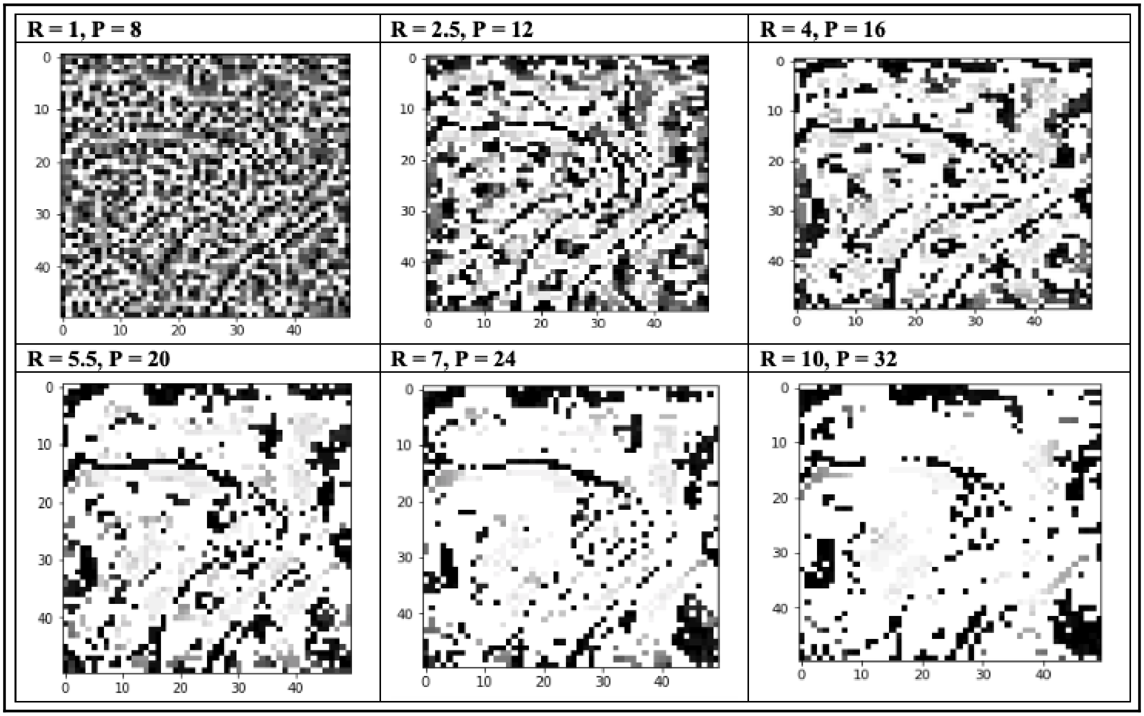




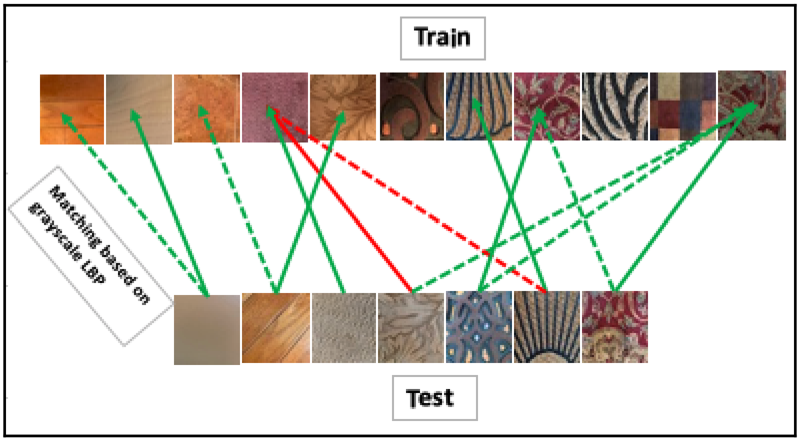
120 119 130
51 82 108
15 9 19
1 1 1
0 1
0 0 0
1 2 4
128 8
64 32 16
1 2 4
0 15 8
0 0 0



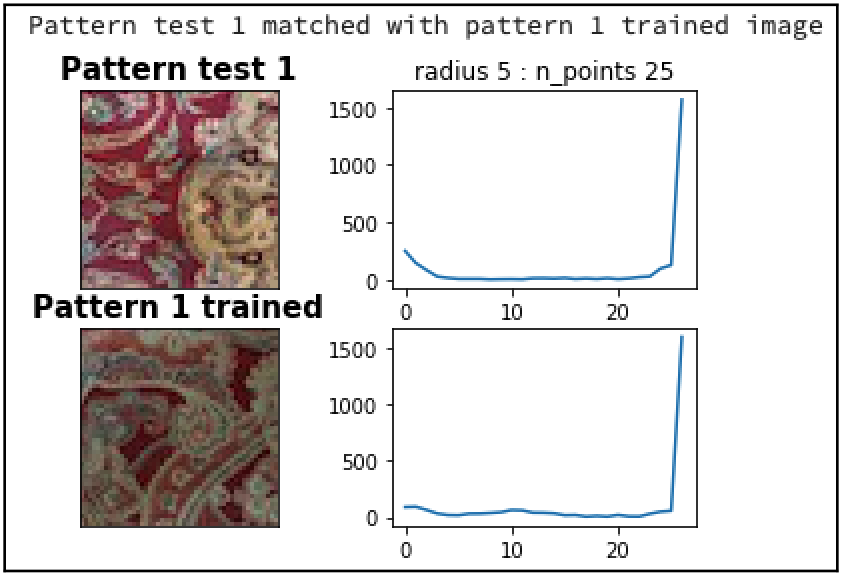
RGB Image
Grayscale Image
LBP Image R=5, P = 25

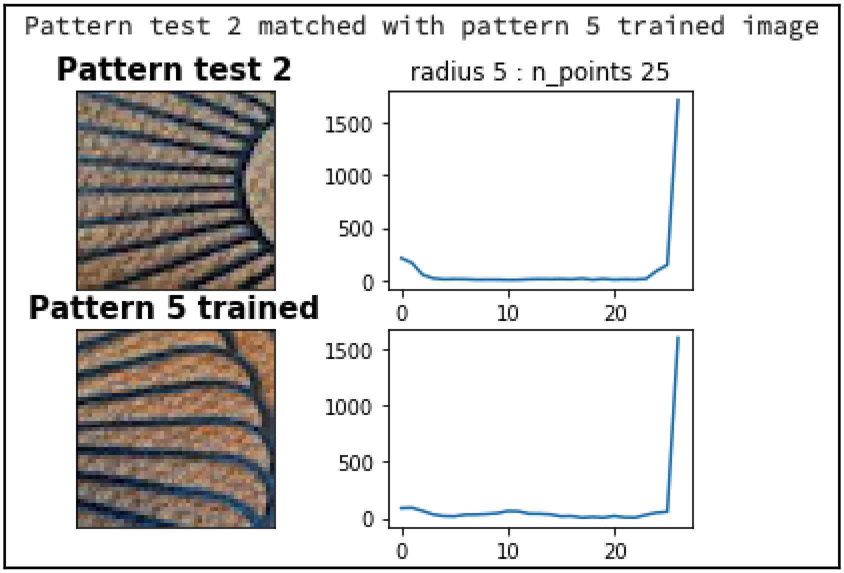


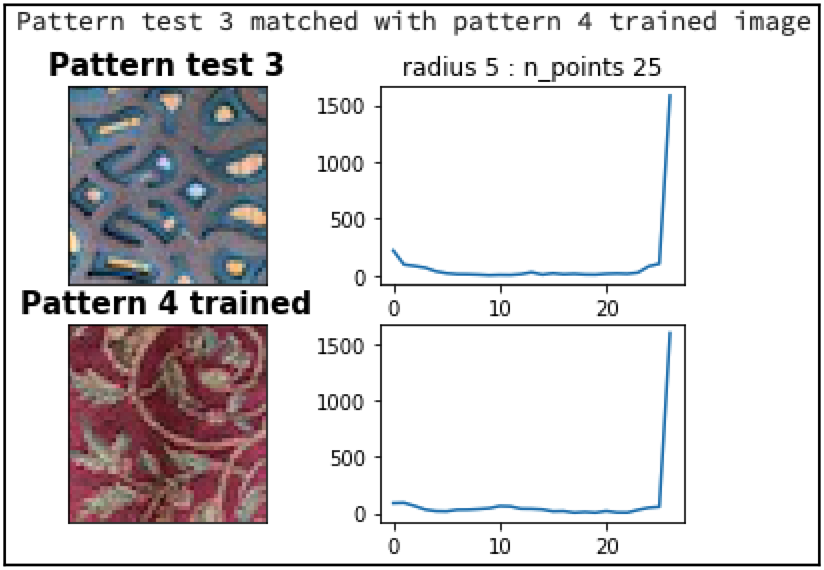
R = 1, P = 8
R = 2.5, P = 12
R = 4, P = 16
R = 5.5, P = 20
R = 7, P = 24
R = 10, P = 32

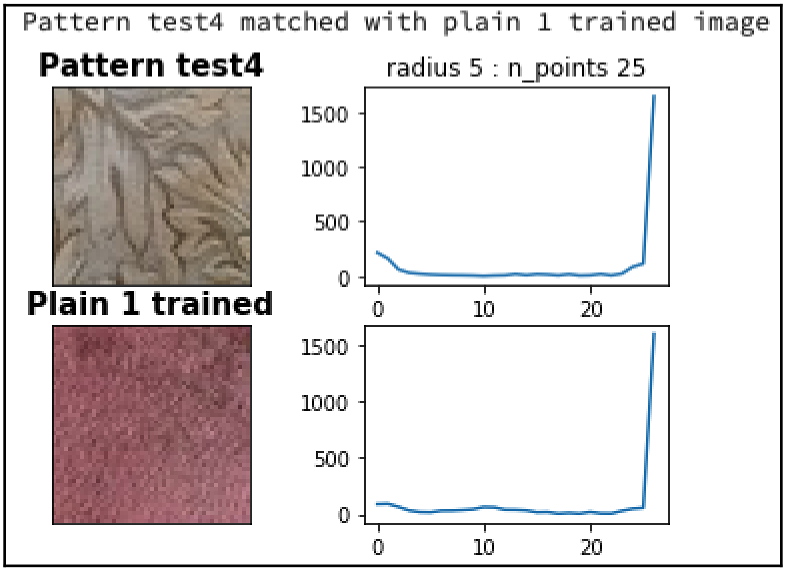


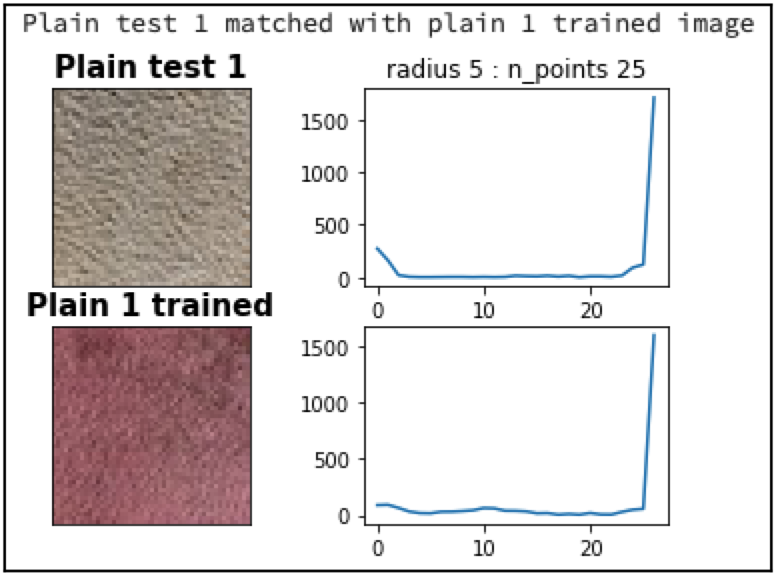
Train
Matching based on grayscale LBP
Test

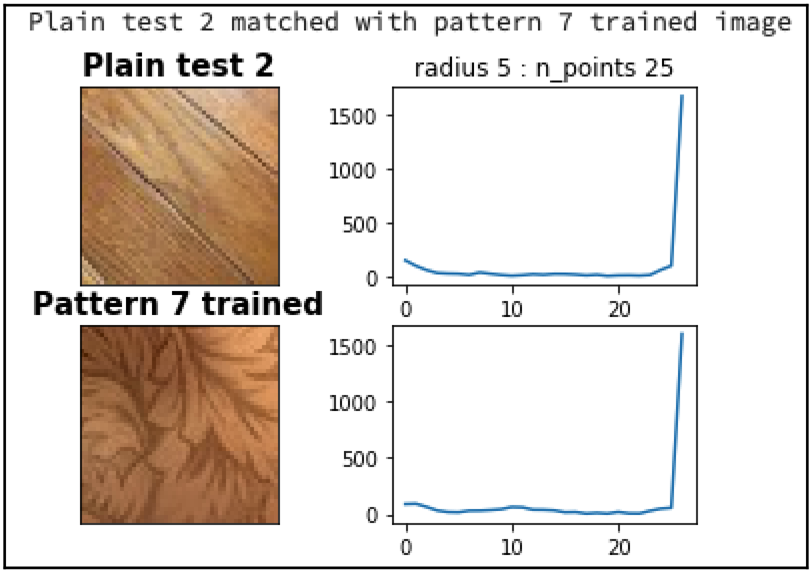

Pattern test 1 matched with pattern 1 trained image
Pattern test 1
radius 5 : n_points 25
Pattern 1 trained
1500
1000
500
0
0
10
20

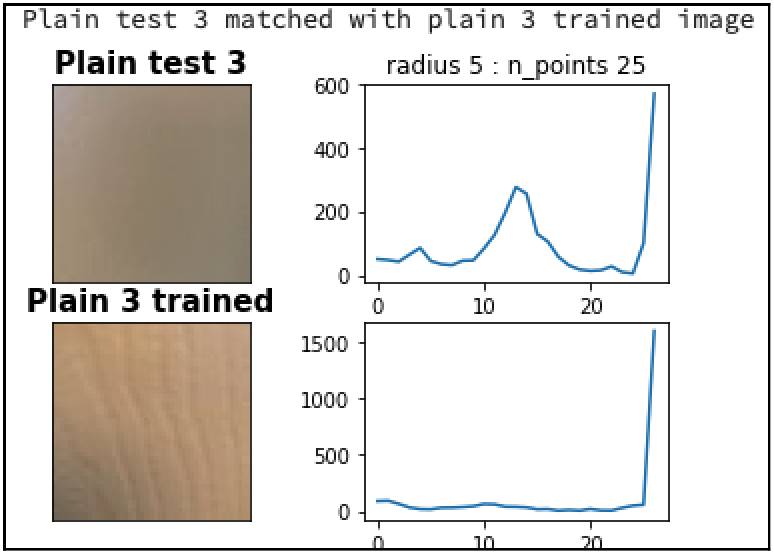

Pattern test 2 matched with pattern 5 trained image
Pattern test 2
radius 5 : n_points 25
Pattern 5 trained
1500
1000
500
0
0
10
20


Pattern test 3 matched with pattern 4 trained image
Pattern test 3
radius 5 : n_points 25
Pattern 4 trained
1500
1000
500
0
0
10
20

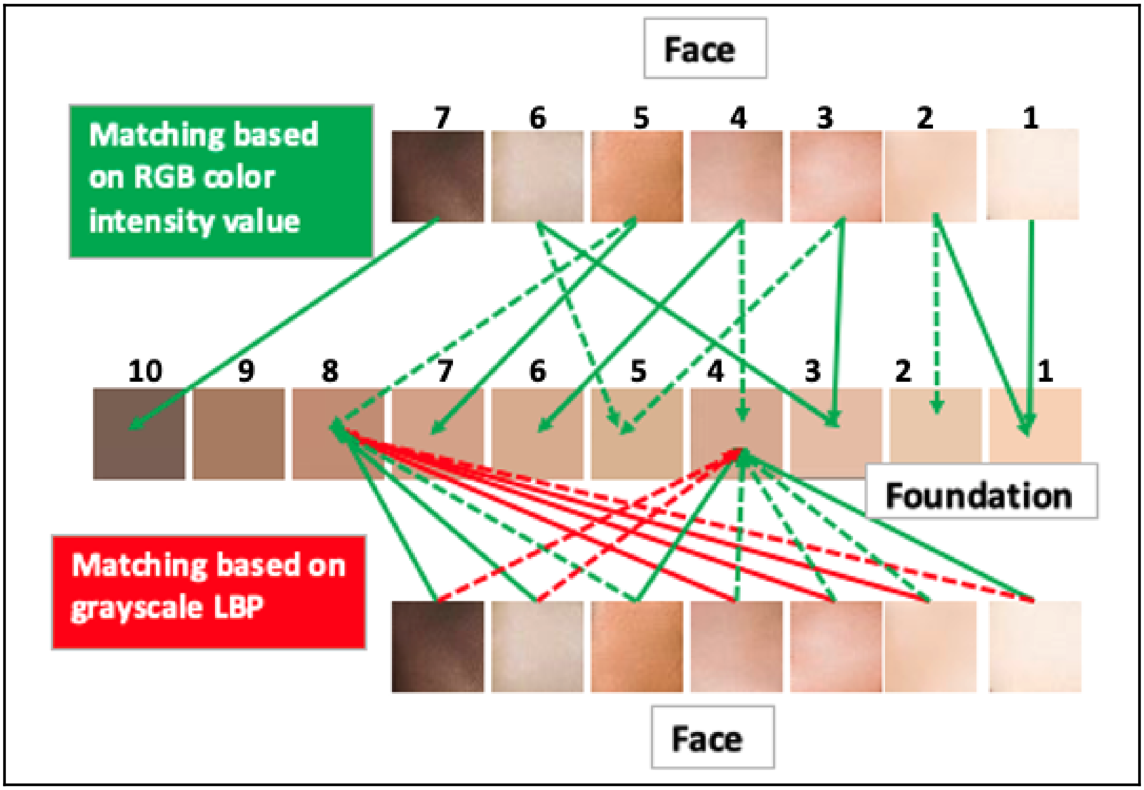

Pattern test4 matched with plain 1 trained image
Pattern test4
radius 5 : n_points 25
Plain 1 trained
1500
1000
500
0
0
10
20


Plain test 1 matched with plain 1 trained image
Plain test 1
Plain 1 trained
radius 5 : n_points 25
1500
1000
500
0
0
10
20

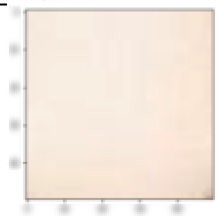
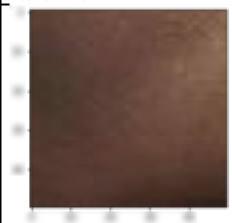
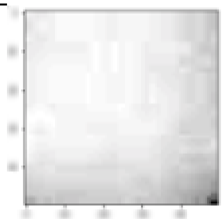
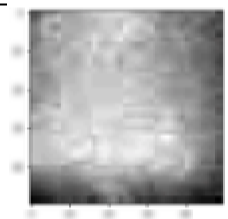
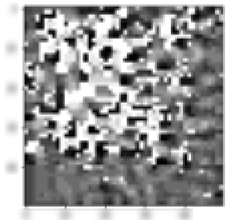
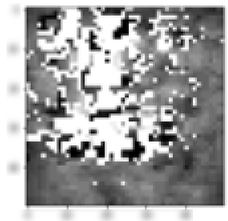

Plain test 2 matched with pattern 7 trained image
Plain test 2
Pattern 7 trained
radius 5 : n_points 25
1500
1000
500
0
0
10
20

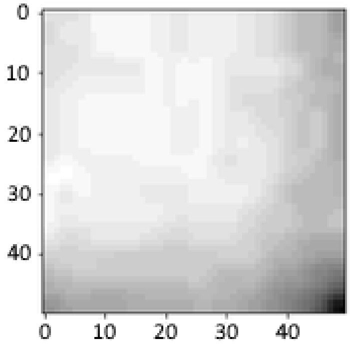
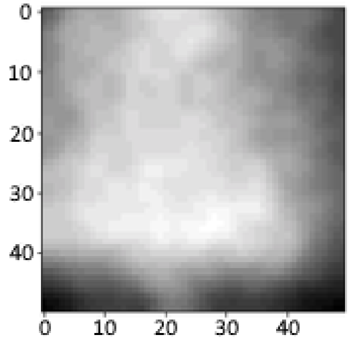
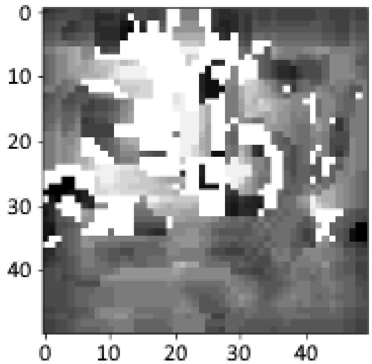
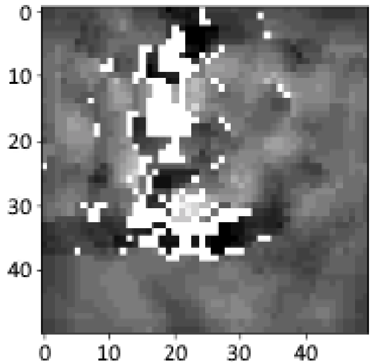


Plain test 3 matched with plain 3 trained image
Plain test 3
radius 5 : n_points 25
600
400
200
0
0
10
20
Plain 3 trained
1500
1000
500
0
0
10
20



Face
Matching based on RGB color intensity value
7
6
5
4
3
2
1
10
9
8
7
6
5
4
3
2
1
Foundation
Matching based on grayscale LBP
Face

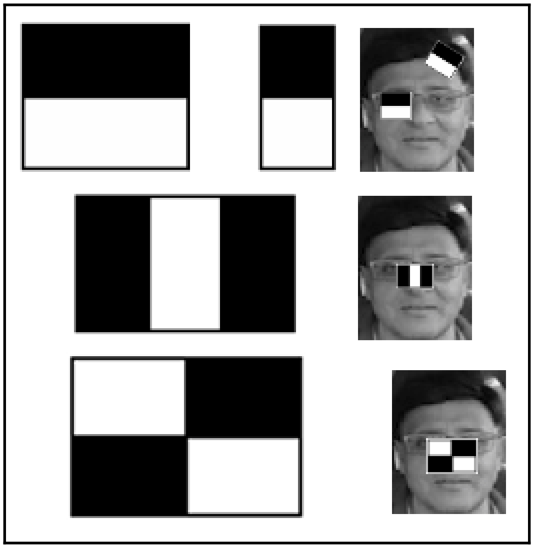
	Face Color 1	Face Color 7
RGB image		
Grayscale image		
LBP image R=2.5, P=16		
LBP image R =5.5, P=20		

	Face color 1	Face color 7
Gray image after applying Gaussian filter 3 times		
LBP image R =5.5, P=20		

$$\Delta I(face(i), found(j)) = 0.299 * (IfaceR - IfoundR) + 0.587 * (IfaceG - IfoundG) + 0.114 * (IfaceB - IfoundB)$$

		Face color						
		1	2	3	4	5	6	7
Foundation color	1	18.4	1.9	19.6	32.2	48.3	18.1	135.8
	2	27.6	7.3	10.4	23.0	39.1	8.9	126.6
	3	36.5	16.3	1.5	14.1	30.2	0.1	117.7
	4	51.6	31.4	13.6	1.0	15.1	15.1	102.6
	5	46.3	26.0	8.3	4.3	20.5	9.8	107.9
	6	51.2	31.0	13.2	0.6	15.5	14.7	103.0
	7	58.3	38.1	20.3	7.7	8.4	21.9	95.9
	8	77.4	57.2	39.4	26.8	10.7	40.9	76.8
	9	95.6	75.4	57.6	45.0	28.9	59.2	58.6
	10	127.2	107.0	89.2	76.6	60.5	90.8	27.0

Chapter 3: Facial Detection Using OpenCV and CNN



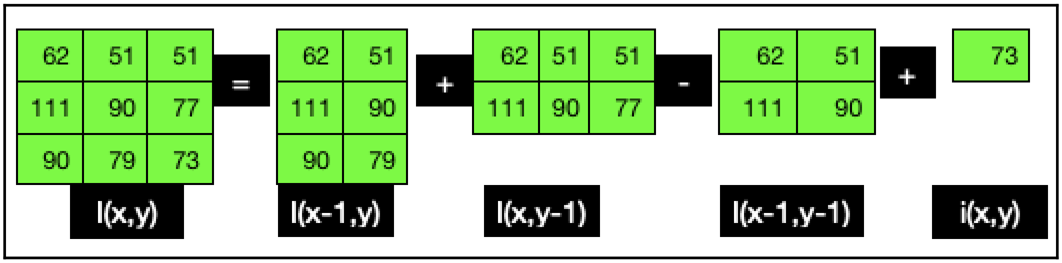


IMAGE PIXEL VALUE

62	51	51	50	54	59	65
111	90	77	64	64	64	61
90	79	73	68	79	99	111
10	14	9	18	46	72	106
53	23	28	15	11	38	61
70	25	11	45	33	19	55
90	69	49	44	27	31	61

INTEGRAL IMAGE PIXEL VALUE

684 = 62+51+51+77+73 +79+90+111+90

866 = 684+50 +64+68

1063 = 866+54 +64+79

917 = 866+10+14 +9+18

62	113	164	214	268	327	392
173	314	442	556	674	797	923
263	483	684	866	1063	1285	1522
273	507	717	917	1160	1454	1797
326	583	821	1036	1290	1622	2026
396	678	927	1187	1474	1825	2284
486	837	1135	1439	1753	2135	2655

1160 = 1063+10+1 4+9+18+46

717 = 684+10+1 4+9

821 = 717+53 +23+28

1036 = 821+50+64+ 68+18+15

1290 = 1036+54+64 +79+46+11

IMAGE PIXEL VALUE

62	51	51	50	54	59	65
111	90	77	64	64	64	61
90	79	73	68	79	99	111
10	14	9	18	46	72	106
53	23	28	15	11	38	61
70	25	11	45	33	19	55
90	69	49	44	27	31	61

INTEGRAL IMAGE PIXEL VALUE

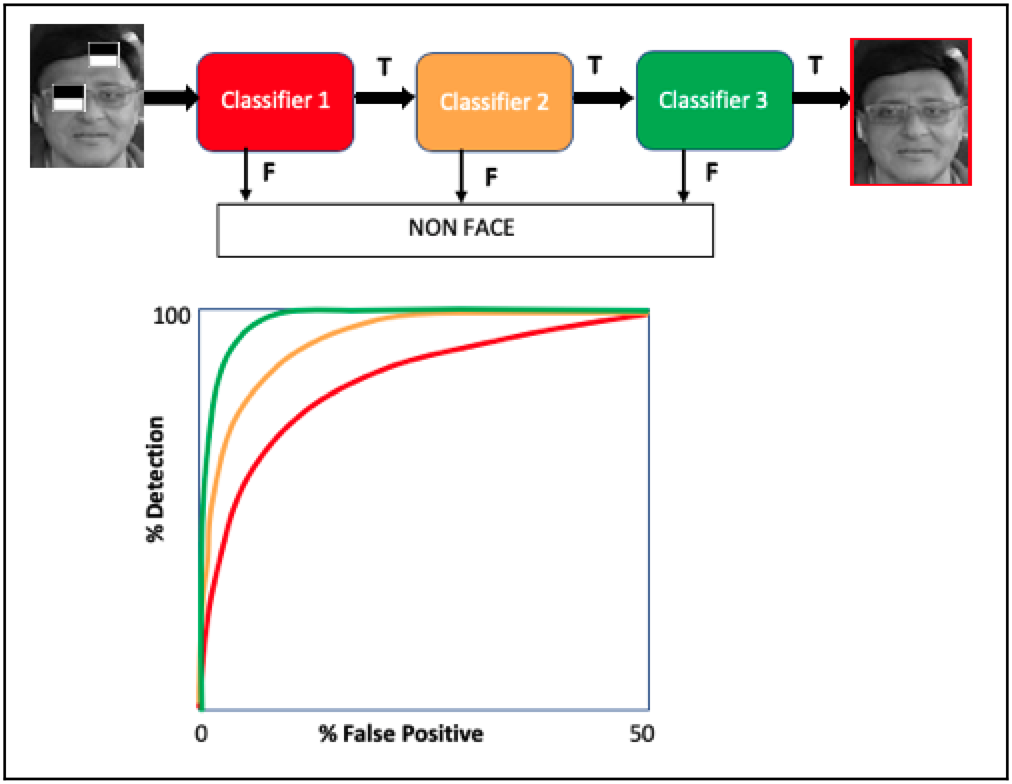
62	113	164	214	268	327	392
173	314	442	556	674	797	923
263	483	684	866	1063	1285	1522
273	507	717	917	1160	1454	1797
326	583	821	1036	1290	1622	2026
396	678	927	1187	1474	1825	2284
486	837	1135	1439	1753	2135	2655

SUM of Haar features = 77 +64 + 64 + 73 + 68 + 79 = 425

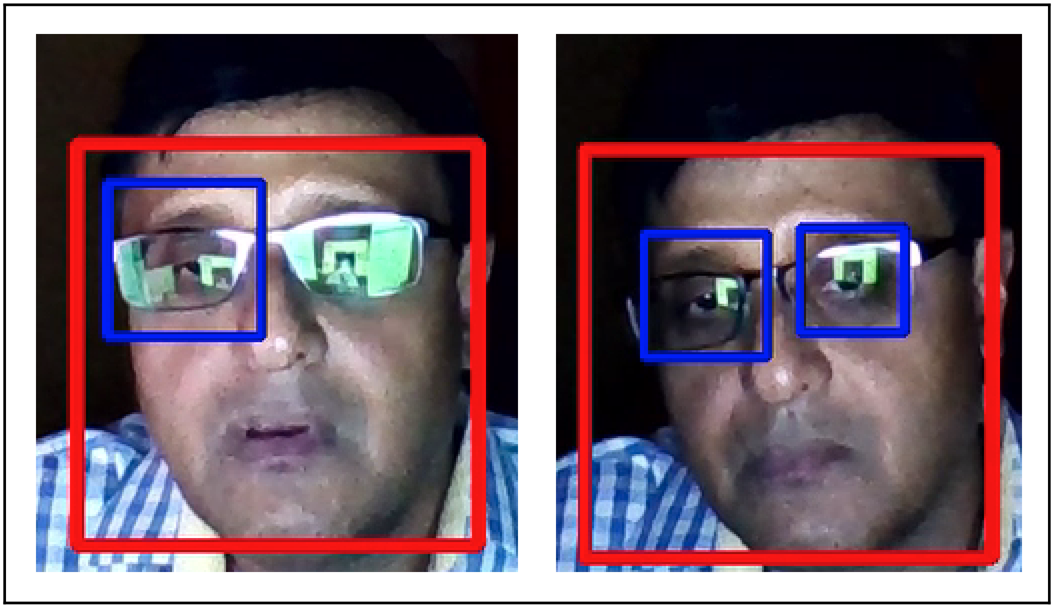
SUM of Haar features using integral image = (1063-268) - (483-113) = 425

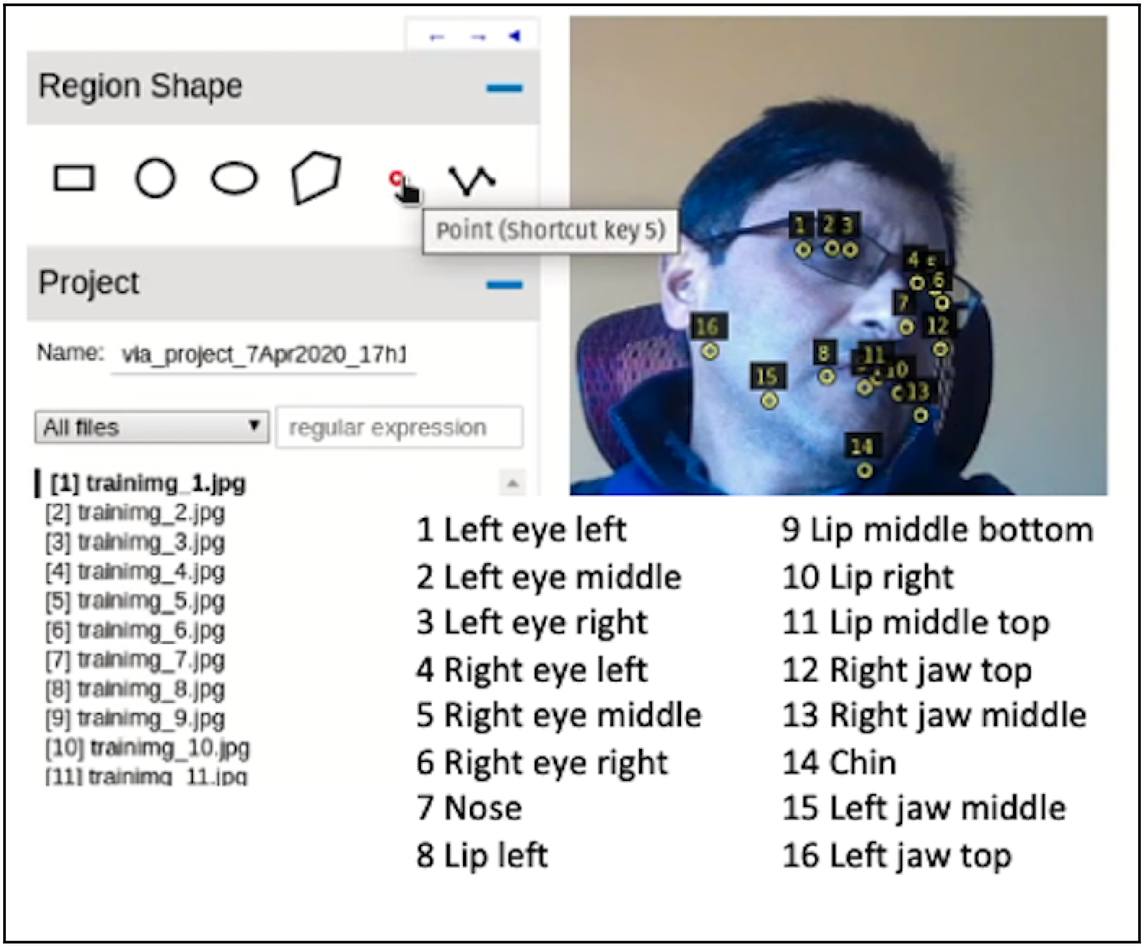
SUM of Haar features = 9 +18 + 46 + 28 + 15 + 11 = 127

SUM of Haar features using integral image = (1290-1063)-(583-483) = 127

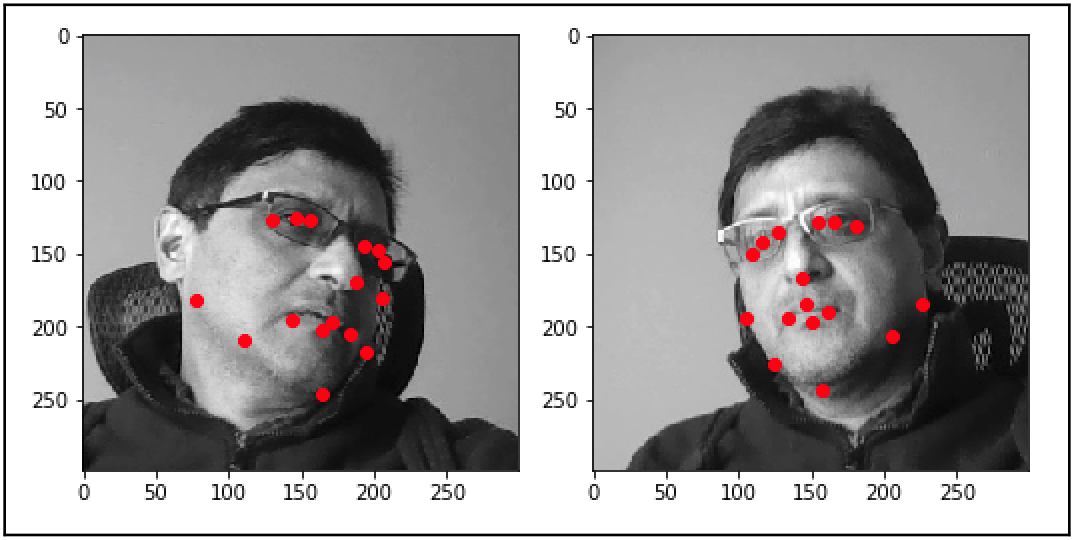


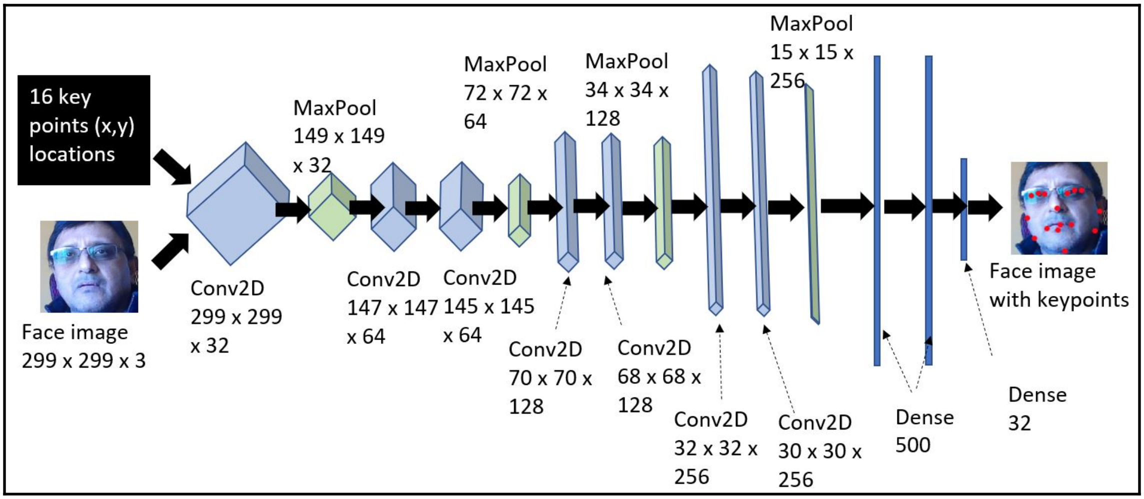
Classifier 1
T
Classifier 2
T
Classifier 3
T
F
F
F
NON FACE
100
% Detection
0
% False Positive
50

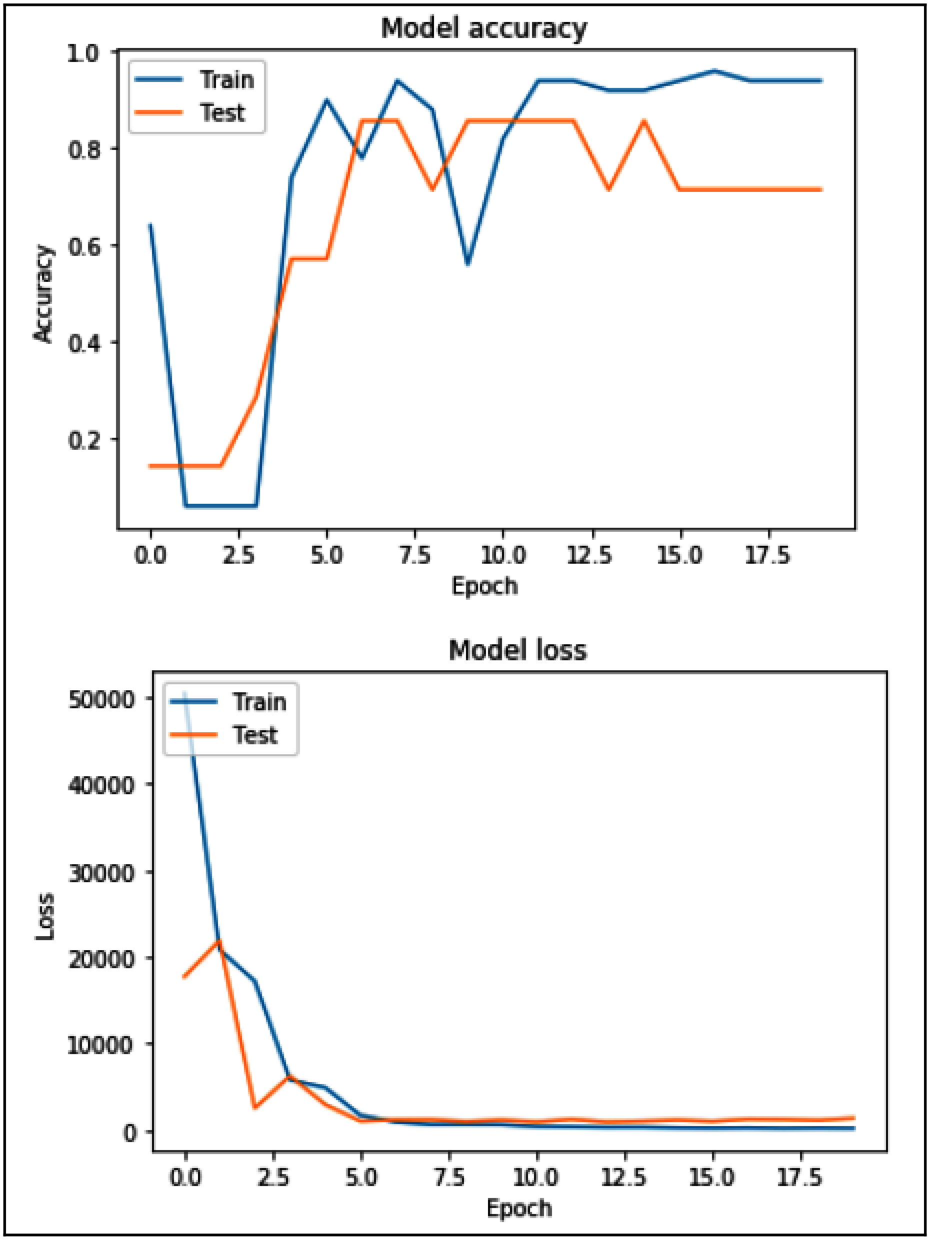


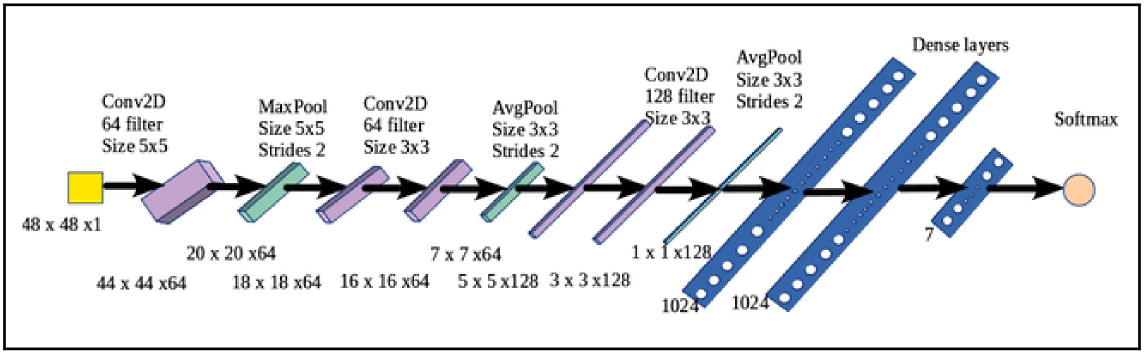


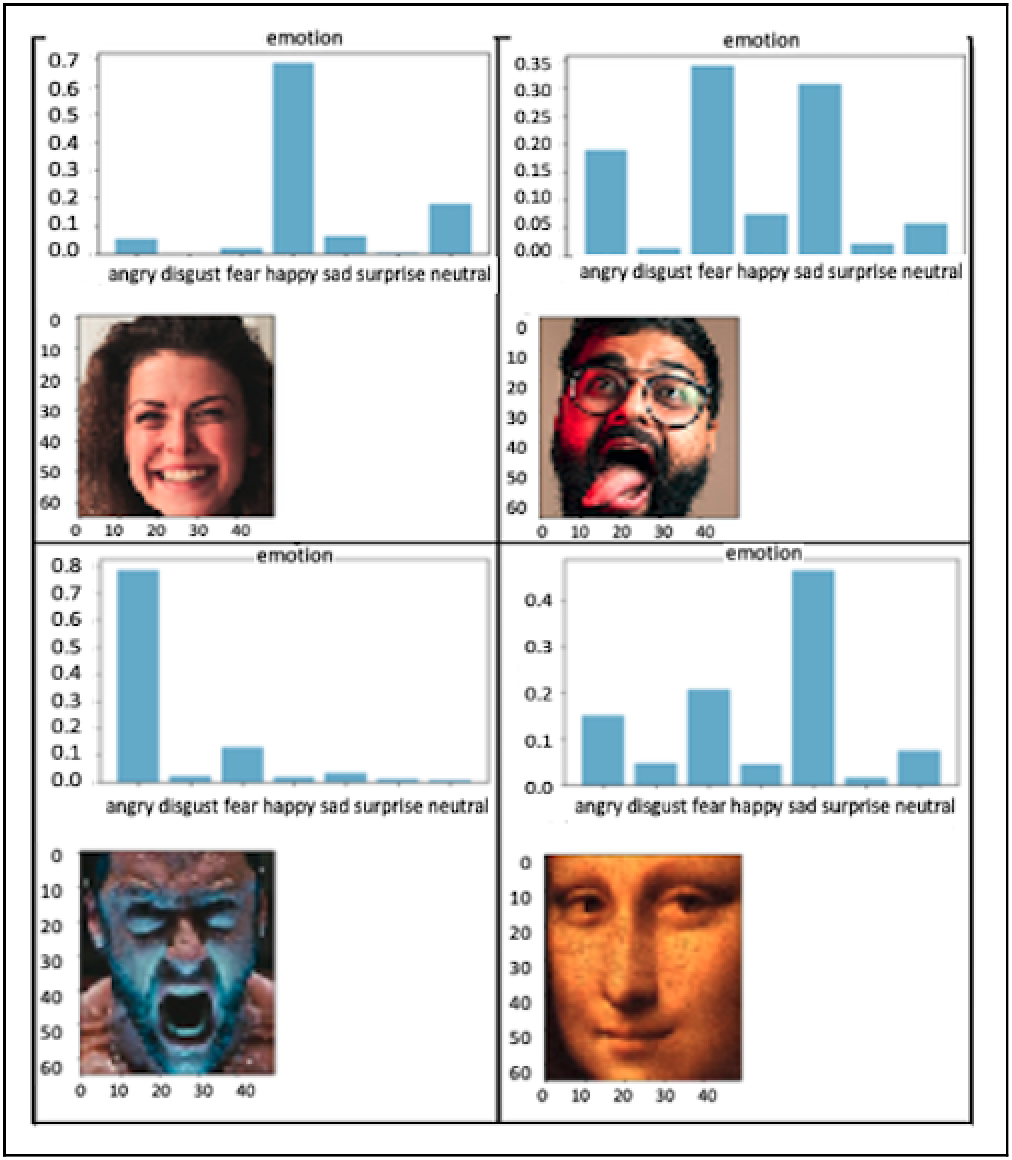
Region Shape
Point (Shortcut key 5)
Project
Name: via_project_7Apr2020_17h1
All files
regular expression
[1] trainimg_1.jpg
[2] trainimg_2.jpg
[3] trainimg_3.jpg
[4] trainimg_4.jpg
[5] trainimg_5.jpg
[6] trainimg_6.jpg
[7] trainimg_7.jpg
[8] trainimg_8.jpg
[9] trainimg_9.jpg
[10] trainimg_10.jpg
[11] trainimg_11.jpg
1 Left eye left
2 Left eye middle
3 Left eye right
4 Right eye left
5 Right eye middle
6 Right eye right
7 Nose
8 Lip left
9 Lip middle bottom
10 Lip right
11 Lip middle top
12 Right jaw top
13 Right jaw middle
14 Chin
15 Left jaw middle
16 Left jaw top


0
50
100
150
200
250
0
50
100
150
200
250

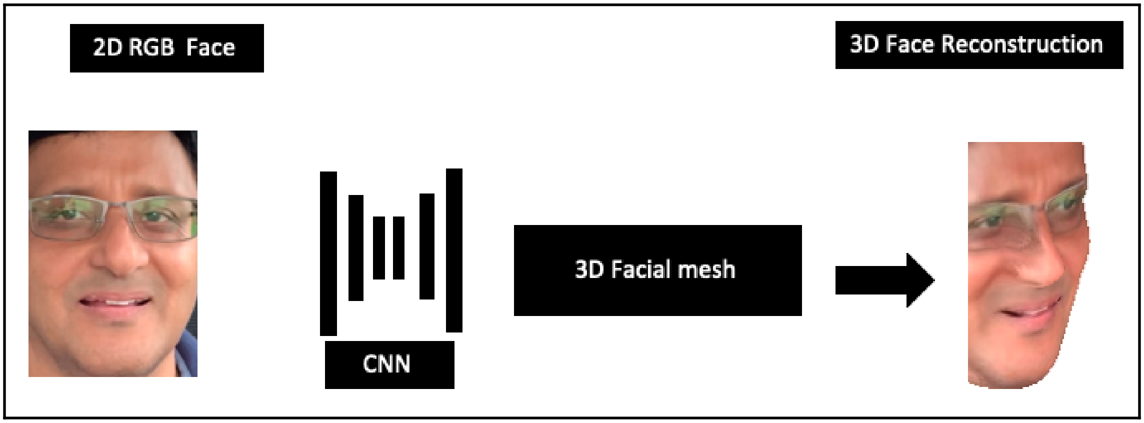

16 key points (x,y) locations
Face image 299 x 299 x 3
Conv2D 299 x 299 x 32
MaxPool 149 x 149 x 32
Conv2D 147 x 147 x 64
Conv2D 145 x 145 x 64
MaxPool 72 x 72 x 64
Conv2D 70 x 70 x 128
Conv2D 68 x 68 x 128
MaxPool 34 x 34 x 128
Conv2D 32 x 32 x 256
Conv2D 30 x 30 x 256
MaxPool 15 x 15 x 256
Dense 500
Dense 32
Face image with keypoints


Model accuracy
Train
Test
Accuracy
Epoch
Model loss
Loss
Epoch

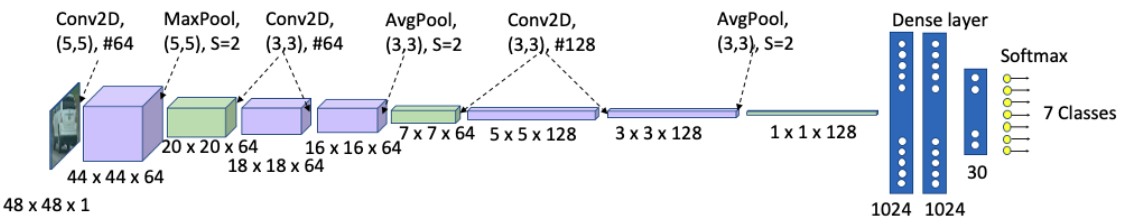

Conv2D
64 filter
Size 5x5
MaxPool
Size 5x5
Strides 2
Conv2D
64 filter
Size 3x3
AvgPool
Size 3x3
Strides 2
Conv2D
128 filter
Size 3x3
AvgPool
Size 3x3
Strides 2
Dense layers
Softmax
48 x 48 x1
20 x 20 x64
7 x 7 x64
1 x 1 x128
44 x 44 x64
18 x 18 x64
16 x 16 x64
5 x 5 x128
3 x 3 x128
1024
1024
7



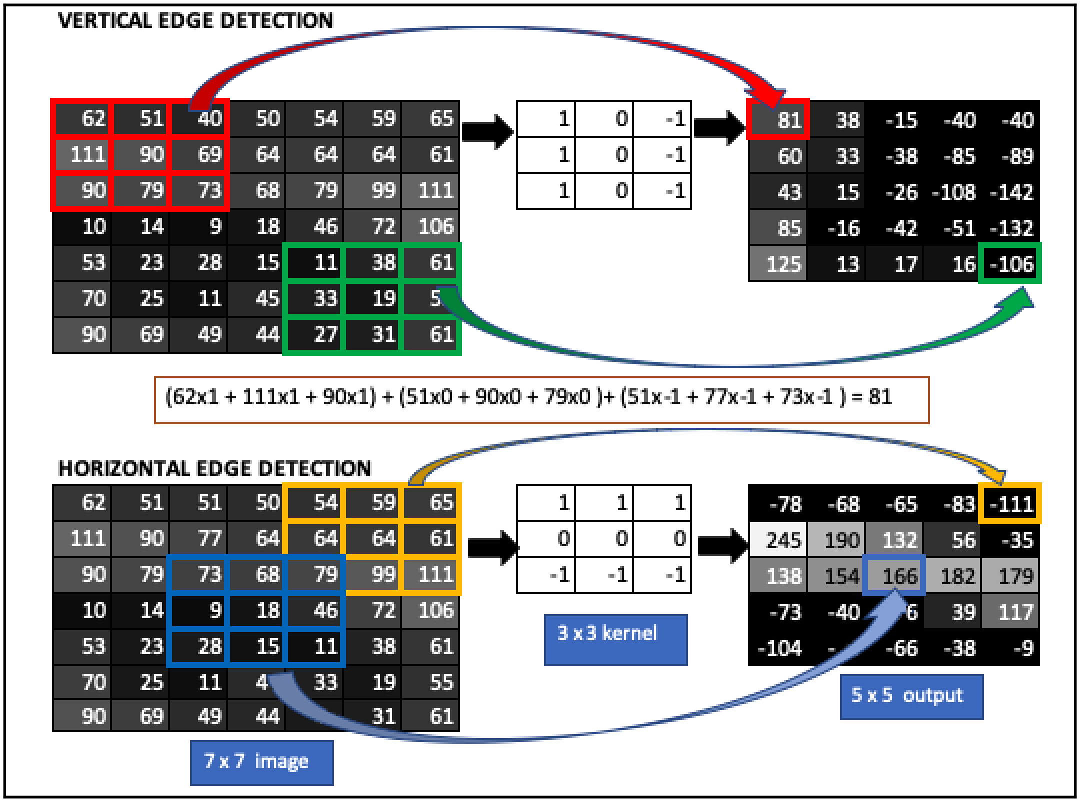
emotion
angry disgust fear happy sad surprise neutral
emotion
angry disgust fear happy sad surprise neutral
emotion
angry disgust fear happy sad surprise neutral
emotion
angry disgust fear happy sad surprise neutral

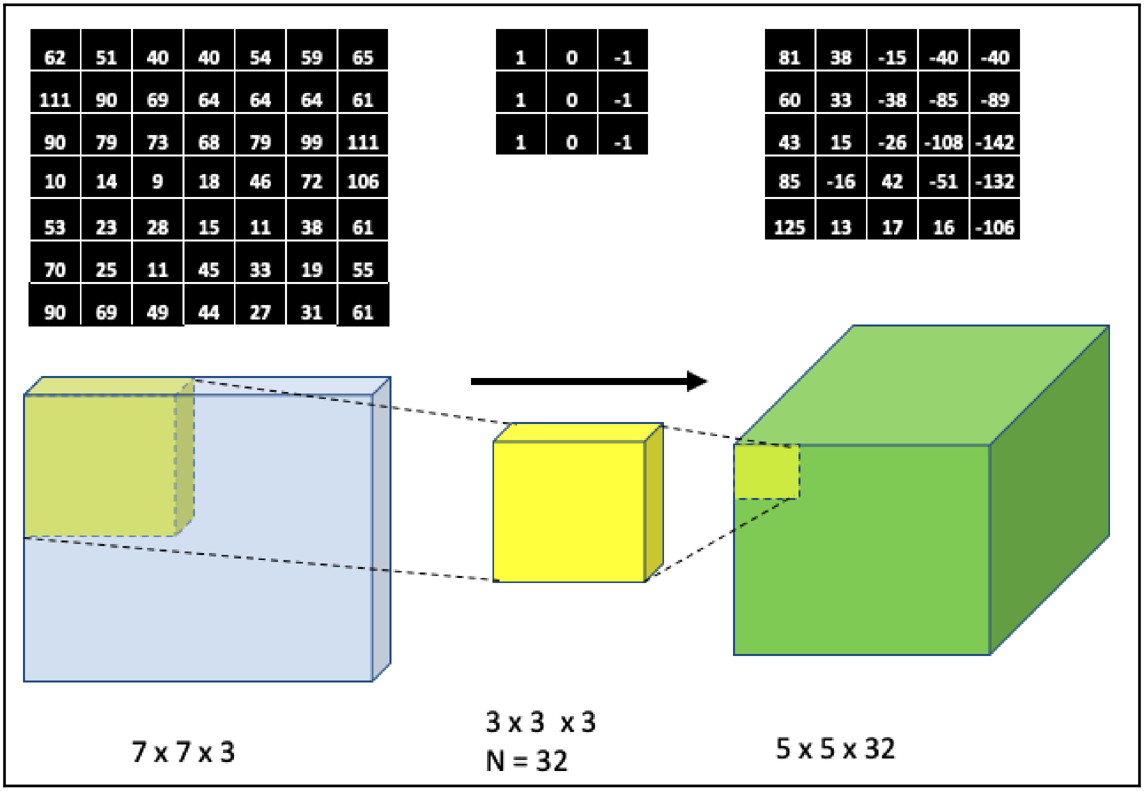

2D RGB Face
3D Face Reconstruction
3D Facial mesh
CNN

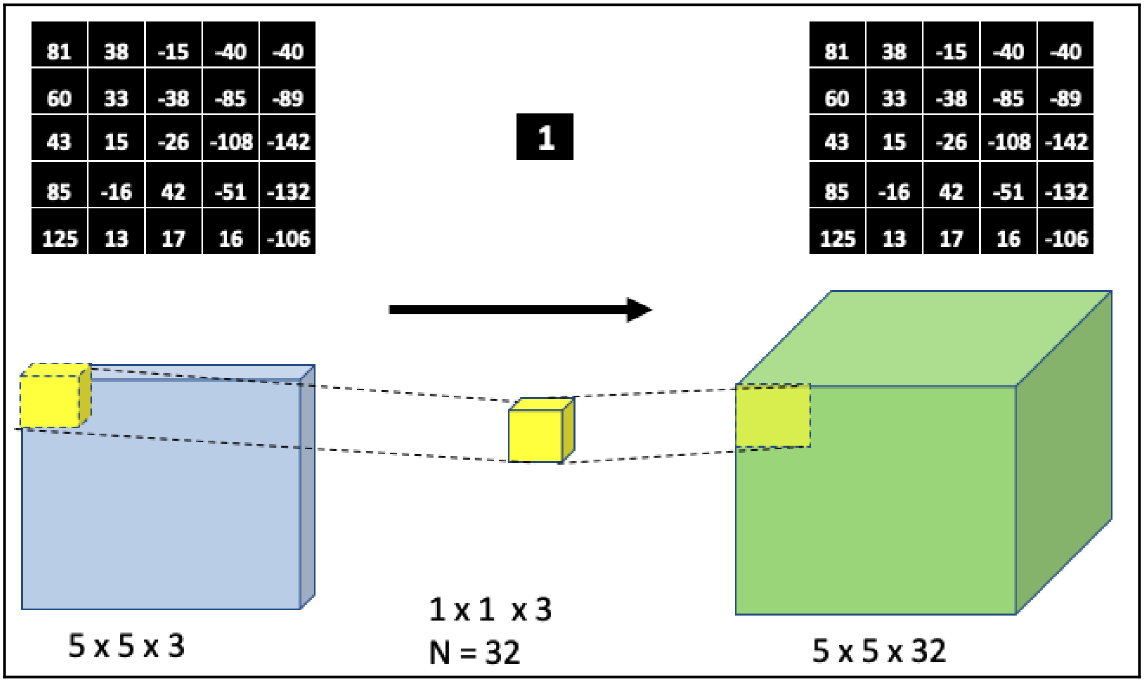
Chapter 4: Deep Learning on Images

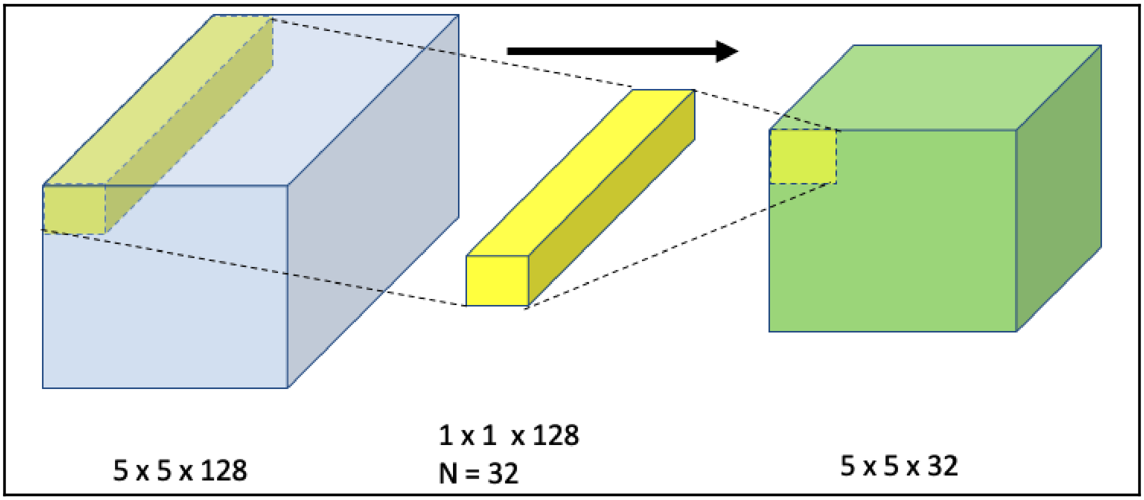


The figure should be read like this:
1) Convolution operation of 64 filters of size (5,5) is applied to a grayscale image of width 48 and height 48 to produce an output shape of 44 x 44 x 64
2) Max pooling operation of filter size (5,5) with stride 2 is applied to the input shape of 44 x 44 x 64. This results in a reduction of output shape to 20 x 20 x 64.
3) A ReLU non-linear activation is applied after the first Conv2D operation and continues after every operation until the last Dense layer.




62 51 40 40 54 59 65
111 90 69 64 64 64 61
90 79 73 68 79 99 111
10 14 9 18 46 72 106
53 23 28 15 11 38 61
70 25 11 45 33 19 55
90 69 49 44 27 31 61
1 0 -1
1 0 -1
1 0 -1
81 38 -15 -40 -40
60 33 -38 -85 -89
43 15 -26 -108 -142
85 -16 42 -51 -132
125 13 17 16 -106
7 x 7 x 3
3 x 3 x 3
N = 32
5 x 5 x 32


81 38 -15 -40 -40
60 33 -38 -85 -89
43 15 -26 -108 -142
85 -16 42 -51 -132
125 13 17 16 -106
1
81 38 -15 -40 -40
60 33 -38 -85 -89
43 15 -26 -108 -142
85 -16 42 -51 -132
125 13 17 16 -106
5 x 5 x 3
1 x 1 x 3
N = 32
5 x 5 x 32


5 x 5 x 128
1 x 1 x 128
N = 32
5 x 5 x 32

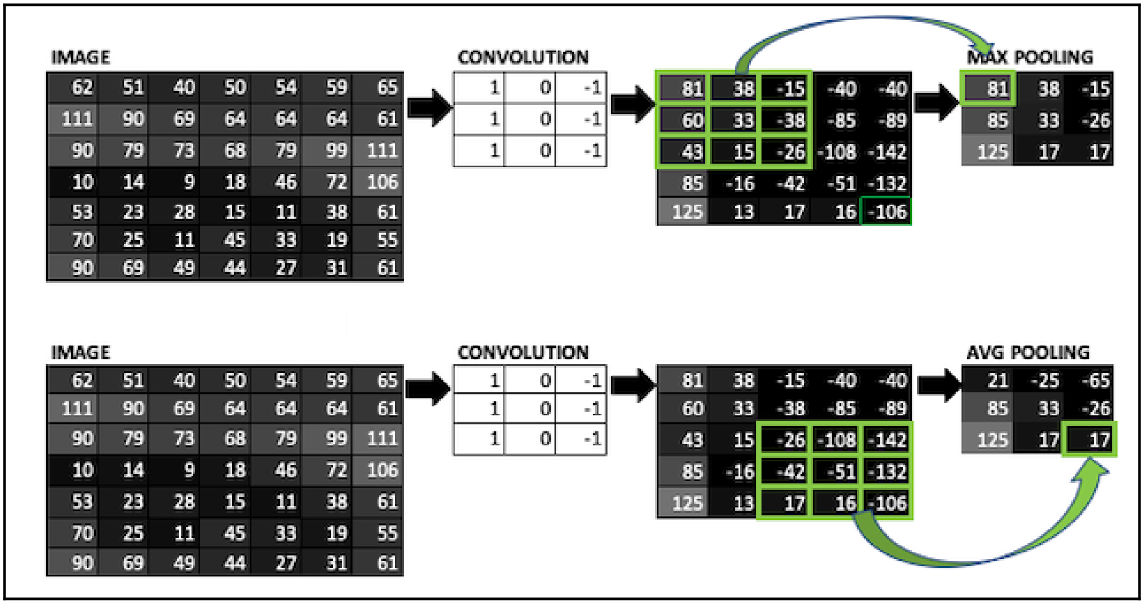
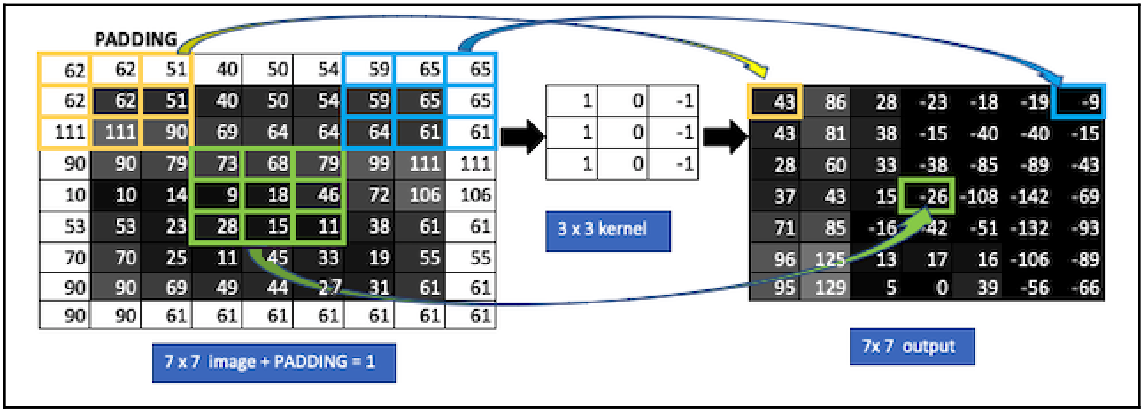
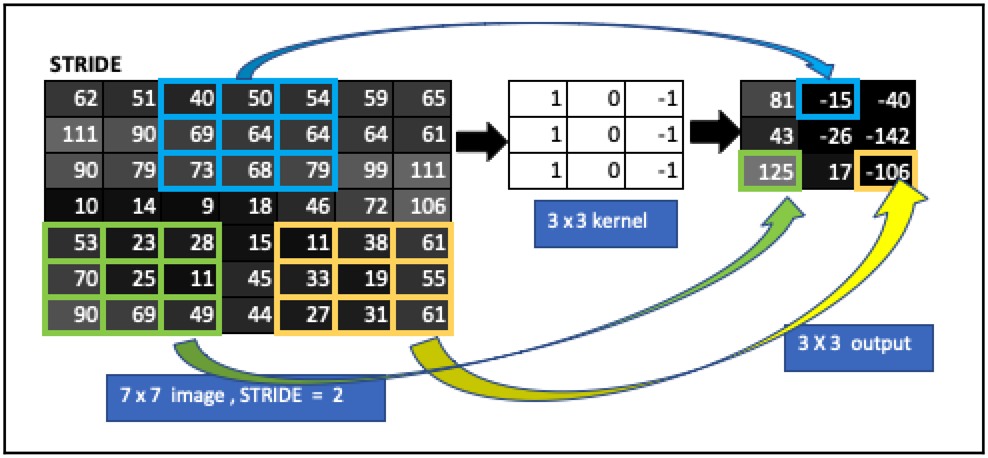
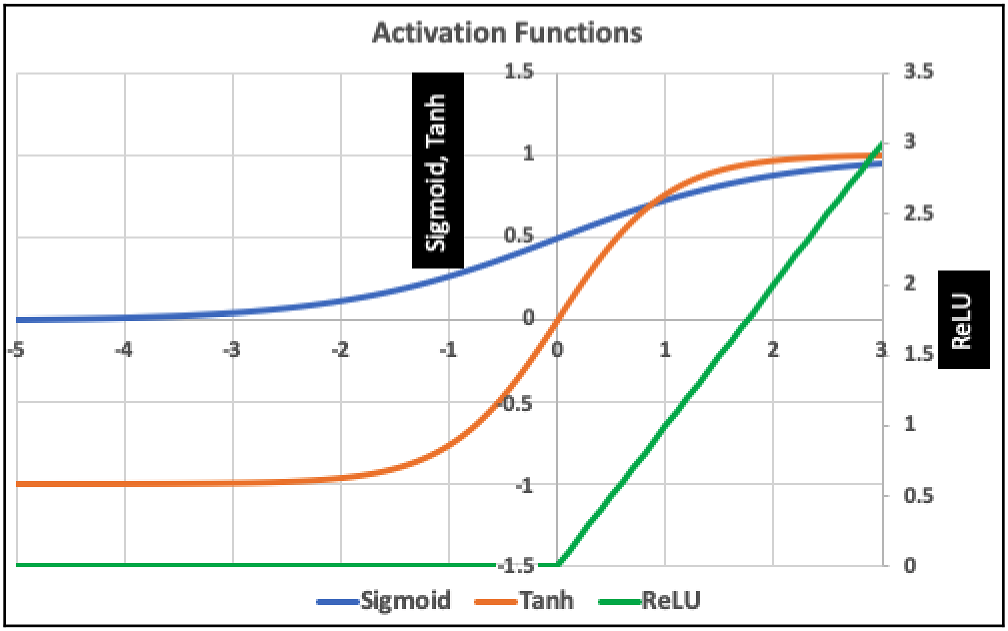
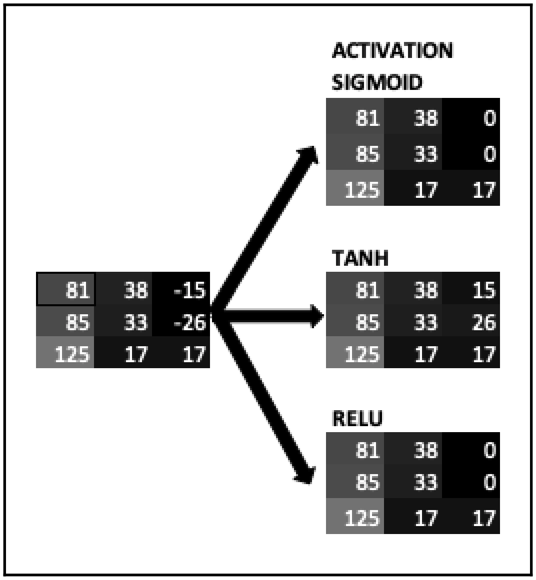


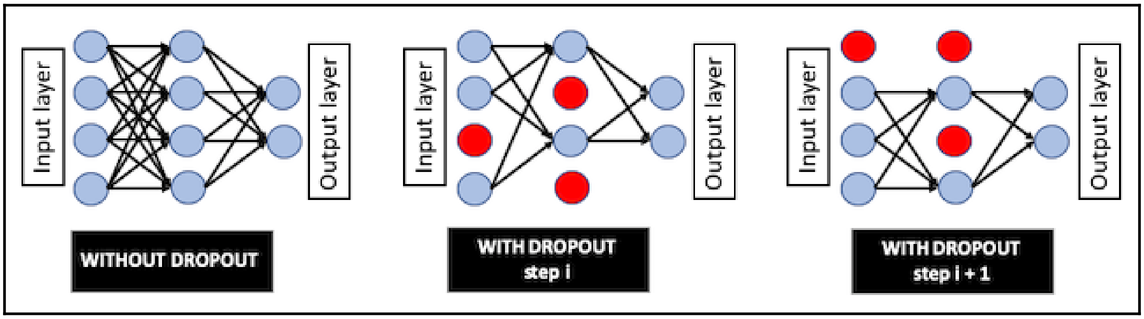
IMAGE
CONVOLUTION
MAX POOLING
IMAGE
CONVOLUTION
AVG POOLING

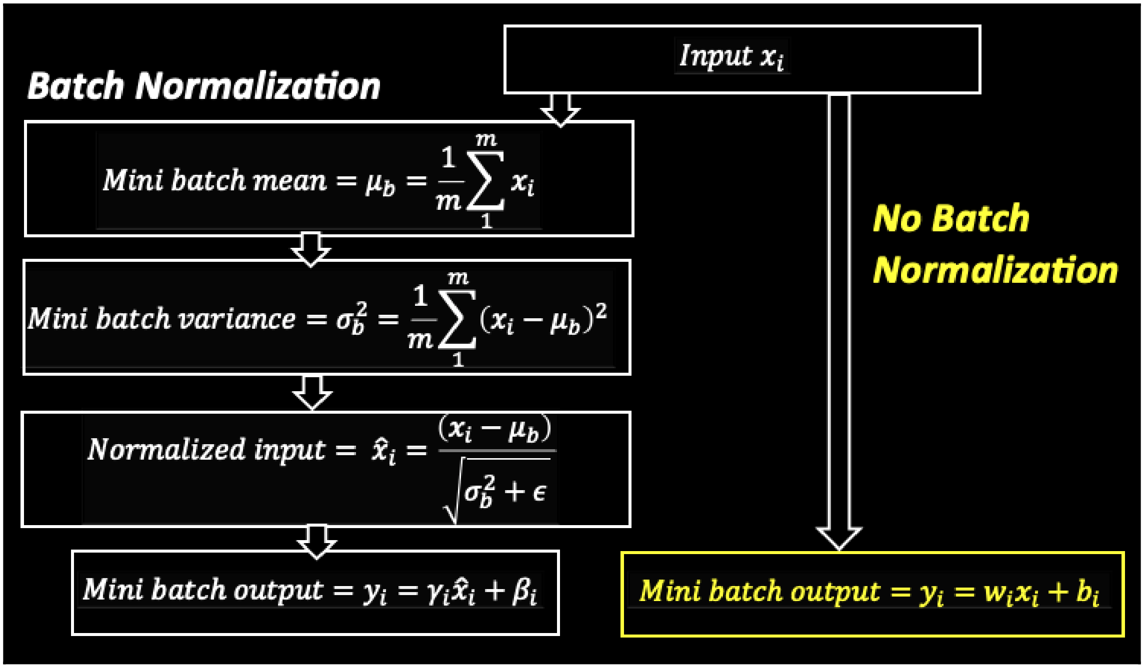

PADDING
3 x 3 kernel
7 x 7 image + PADDING = 1
7x 7 output


STRIDE
62 51 40 50 54 59 65
111 90 69 64 64 64 61
90 79 73 68 79 99 111
10 14 9 18 46 72 106
53 23 28 15 11 38 61
70 25 11 45 33 19 55
90 69 49 44 27 31 61
1 0 -1
1 0 -1
1 0 -1
81 -15 -40
43 -26 -142
125 17 -106
3 x 3 kernel
3 X 3 output
7 x 7 image , STRIDE = 2


Activation Functions
Sigmoid, Tanh
ReLU
1.5
1
0.5
0
-0.5
-1
-1.5
-5
-4
-3
-2
-1
0
1
2
3
3.5
3
2.5
2
1.5
1
0.5
0
Sigmoid
Tanh
ReLU

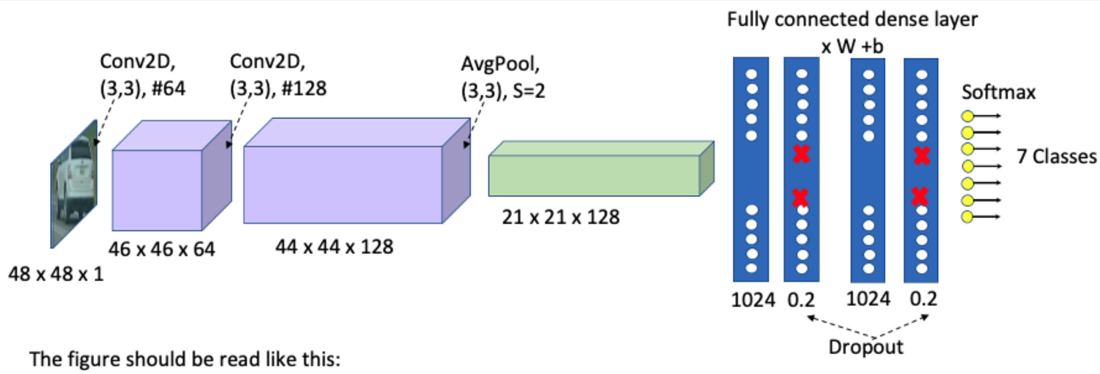

ACTIVATION
SIGMOID
TANH
RELU


Input layer
Output layer
WITHOUT DROPOUT
WITH DROPOUT step i
WITH DROPOUT step i + 1



yi	exp(yi)	Probability = exp(yi)/Σexp(yi)
0.1	1.1	3%
1.5	4.5	12%
0.01	1.0	3%
2	7.4	20%
0.5	1.6	4%
3	20.1	54%
0.25	1.3	3%
SUM	**37.0**	**100%**

Feature map operations	Feature map shape	Feature map size	# of parameters
Input image	(48,48,1)	2304	0
CONV1 (f=5, nf=64, s=1)	(44,44,64)	123904	1664
POOL1(f=5,s=2)	(20,20,64)	25600	0
CONV2 (f=3, nf=64, s=1)	(18,18,64)	20736	640
CONV3 (f=3, nf=64, s=1)	(16,16,64)	16384	640
POOL2(f=3,s=2)	(7,7,64)	3136	0
CONV4 (f=3, nf=128, s=1)	(5,5,128)	3200	1280
CONV5 (f=3, nf=128, s=1)	(3,3,128)	1152	1280
POOL3(f=3,s=2)	(1,1,128)	128	0
FC1 (1024)	(1024,1)	1024	131073
DROP(.2)	(820,1)	820	104961
FC1 (1024)	(1024,1)	1024	1048577
DROP(.2)	(820,1)	820	672401
Softmax	(7,1)	7	5741

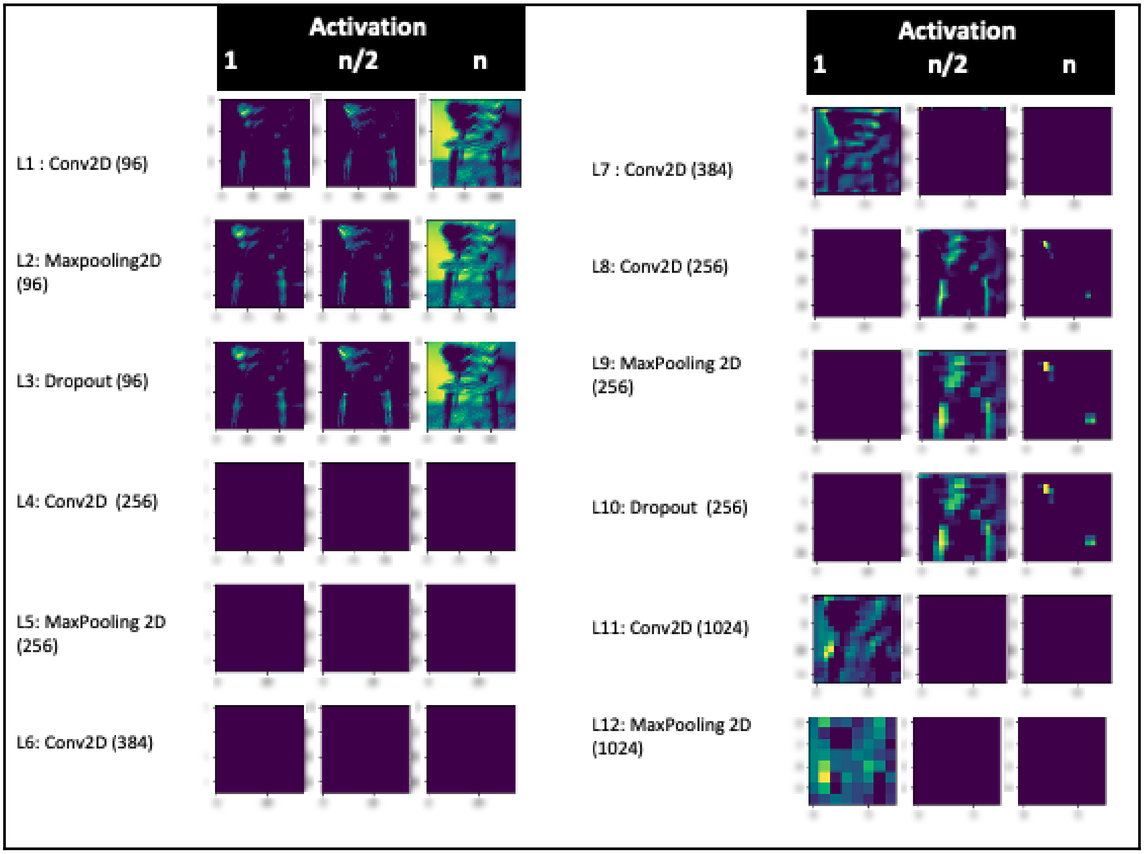


The figure should be read like this:

1) Convolution operation of 64 filters of size (3,3) is applied to a grayscale image of width 48 and height 48 to produce an output shape of 46 x 46 x 64

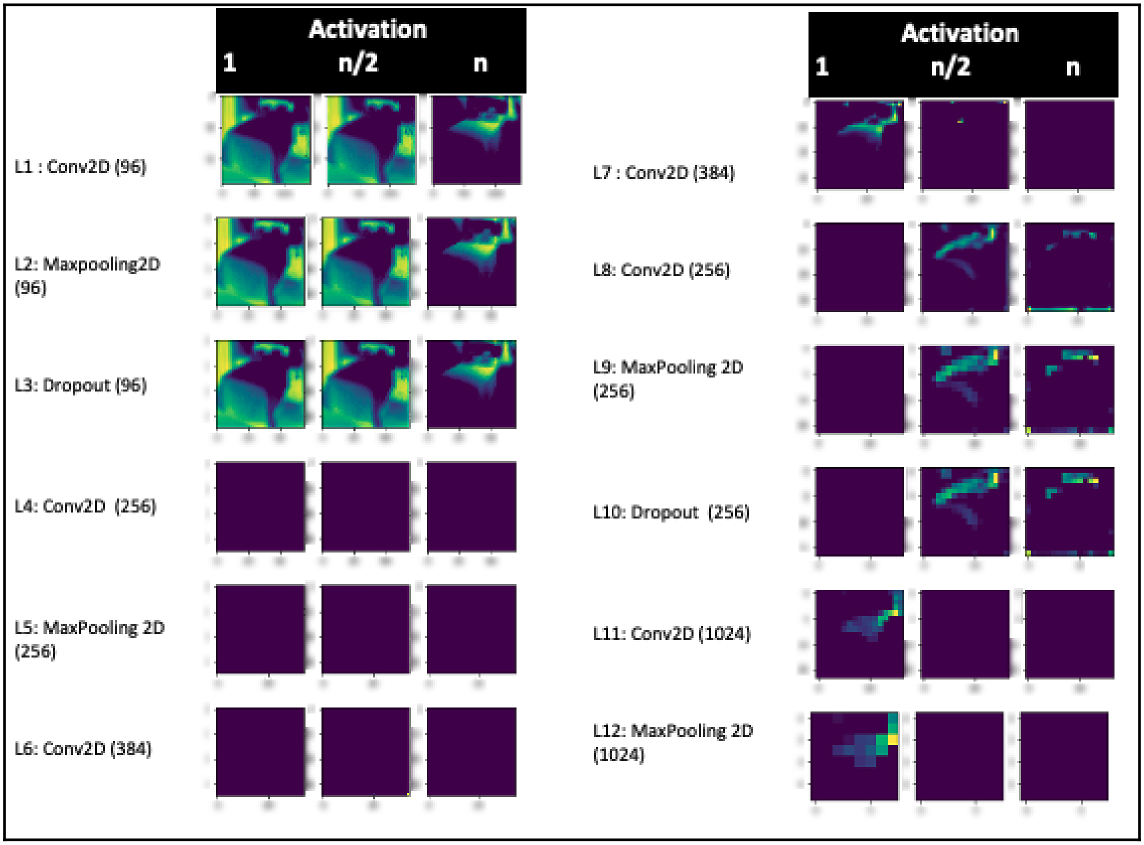
2) Average pooling operation of filter size (3,3) with stride 2 is applied to the input shape of 44 x 44 x 128. This results in a reduction of output shape to 21 x 21 x 128.

3) A ReLU non-linear activation is applied after the first Conv2D operation and continues after every operation until the last Dense layer.

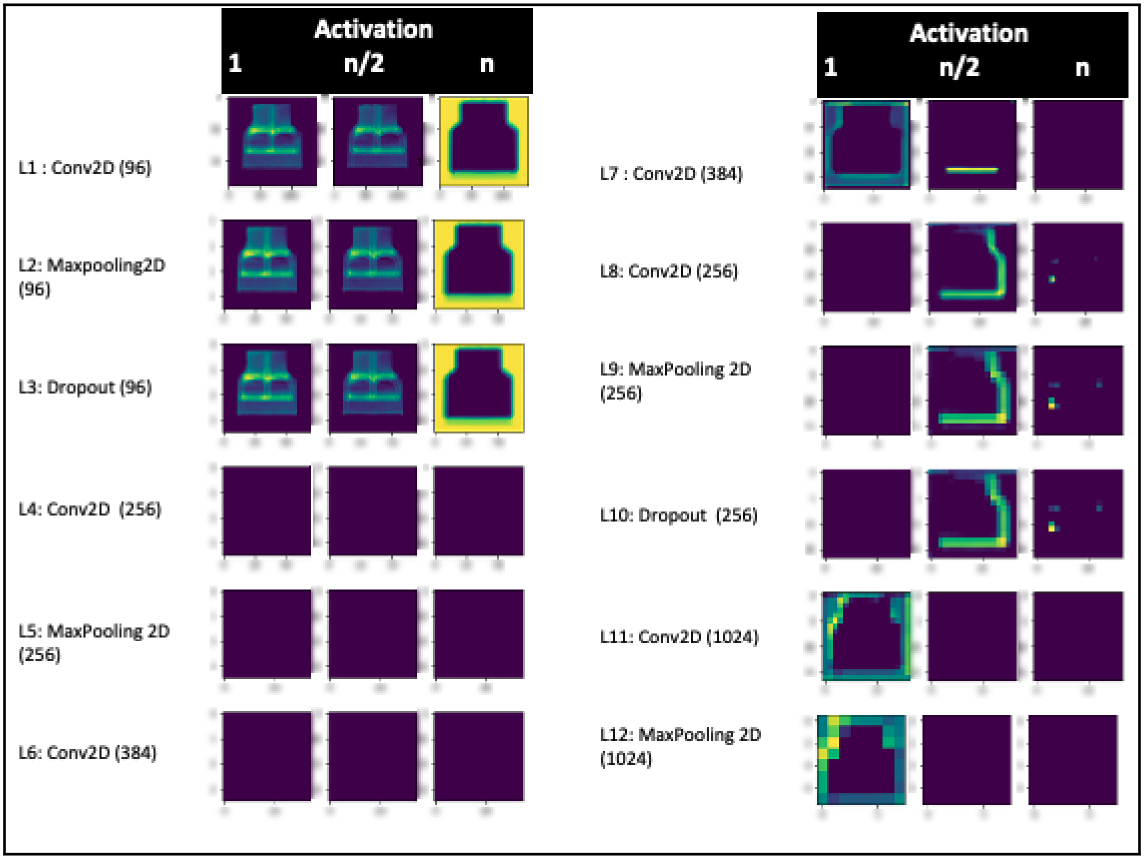


Activation
1
n/2
n
L1 : Conv2D (96)
L2: Maxpooling2D (96)
L3: Dropout (96)
L4: Conv2D (256)
L5: MaxPooling 2D (256)
L6: Conv2D (384)
Activation
1
n/2
n
L7 : Conv2D (384)
L8: Conv2D (256)
L9: MaxPooling 2D (256)
L10: Dropout (256)
L11: Conv2D (1024)
L12: MaxPooling 2D (1024)

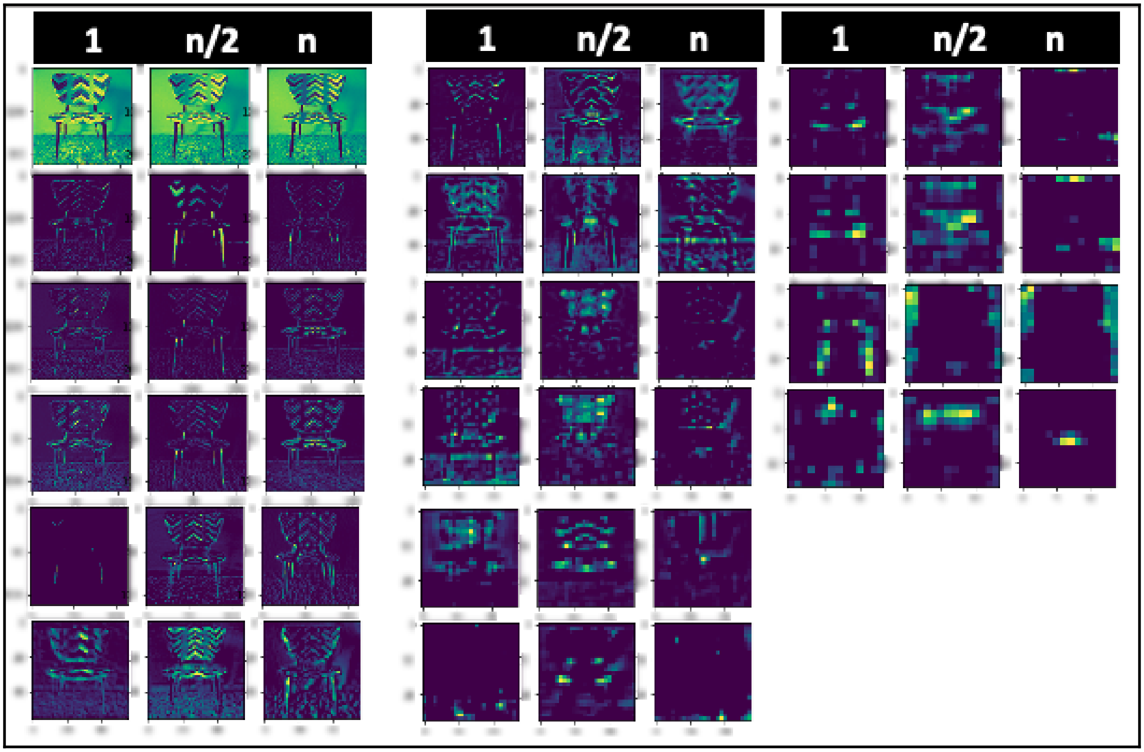
```
Model: "sequential'

Layer (type)              Output Shape              Param #
=================================================================
conv2d (Conv2D)           (None, 140, 140, 96)      3494

max_pooling2d (MaxPooling2D) (None, 70, 70, 96)     0

dropout (Dropout)         (None, 70, 70, 96)        0

conv2d_1 (Conv2D)         (None, 70, 70, 256)       614656

max_pooling2d_1 (MaxPooling2 (None, 35, 35, 256)    0

conv2d_2 (Conv2D)         (None, 35, 35, 384)       885120

conv2d_3 (Conv2D)         (None, 35, 35, 384)       1327488

conv2d_4 (Conv2D)         (None, 35, 35, 256)       884992

max_pooling2d_2 (MaxPooling2 (None, 17, 17, 256)    0

dropout_1 (Dropout)       (None, 17, 17, 256)       0

conv2d_5 (Conv2D)         (None, 17, 17, 1024)      2360320

max_pooling2d_3 (MaxPooling2 (None, 8, 8, 1024)     0

dropout_2 (Dropout)       (None, 8, 8, 1024)        0

flatten (Flatten)         (None, 65536)  0

dense (Dense)             (None, 4096)    268439552

dense_1 (Dense)           (None, 3)       12291
=================================================================
Total params: 274,559,363
Trainable params: 274,559,363
Non-trainable params: 0
```

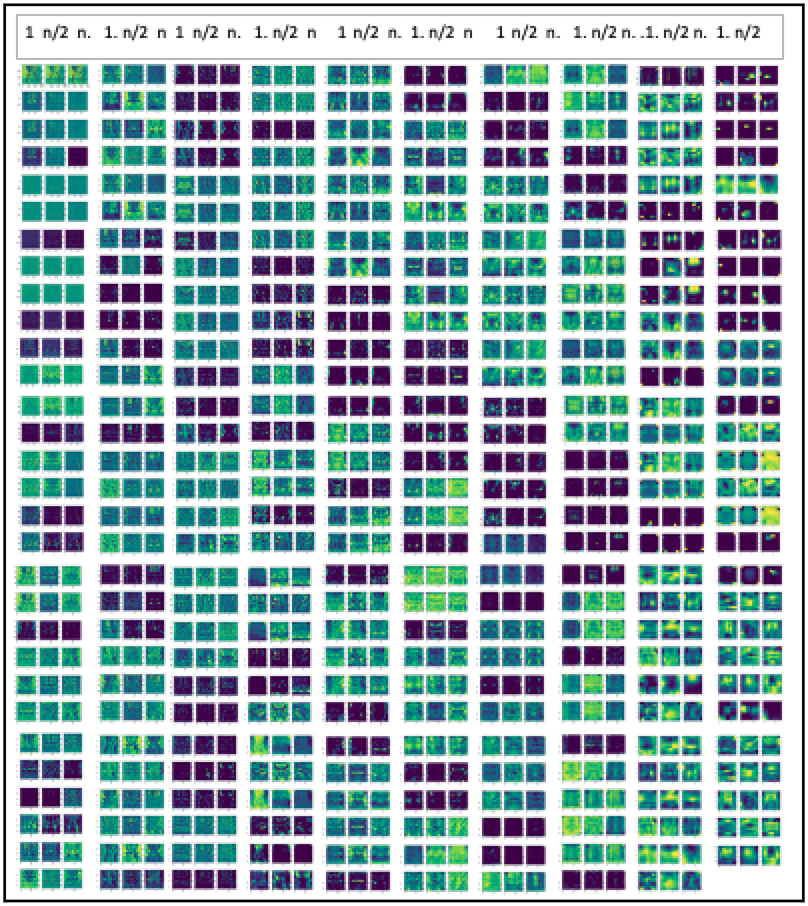


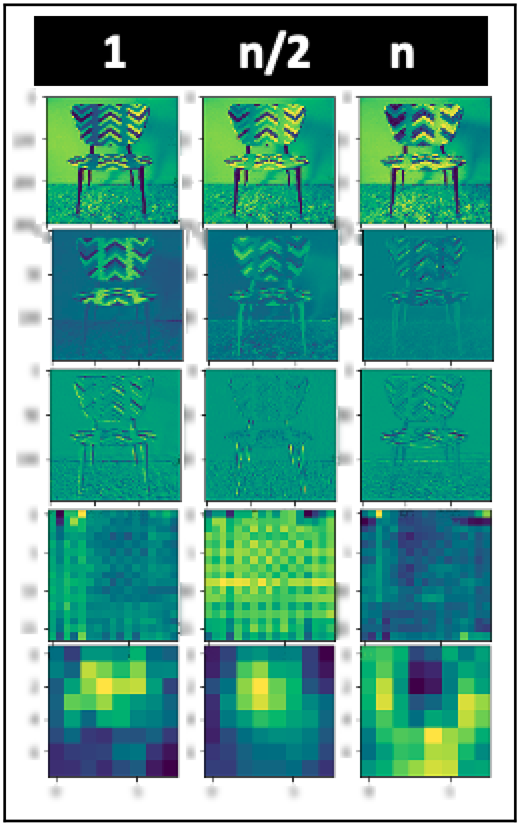
Activation
1
n/2
n
L1 : Conv2D (96)
L2: Maxpooling2D (96)
L3: Dropout (96)
L4: Conv2D (256)
L5: MaxPooling 2D (256)
L6: Conv2D (384)
Activation
1
n/2
n
L7 : Conv2D (384)
L8: Conv2D (256)
L9: MaxPooling 2D (256)
L10: Dropout (256)
L11: Conv2D (1024)
L12: MaxPooling 2D (1024)



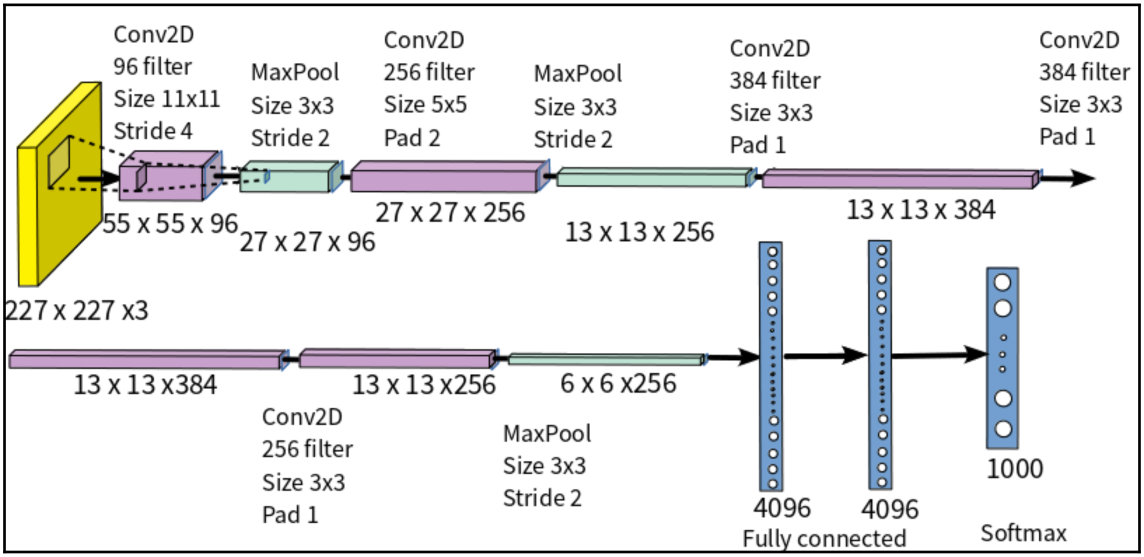
Activation
1
n/2
n
L1 : Conv2D (96)
L2: Maxpooling2D (96)
L3: Dropout (96)
L4: Conv2D (256)
L5: MaxPooling 2D (256)
L6: Conv2D (384)
Activation
1
n/2
n
L7 : Conv2D (384)
L8: Conv2D (256)
L9: MaxPooling 2D (256)
L10: Dropout (256)
L11: Conv2D (1024)
L12: MaxPooling 2D (1024)

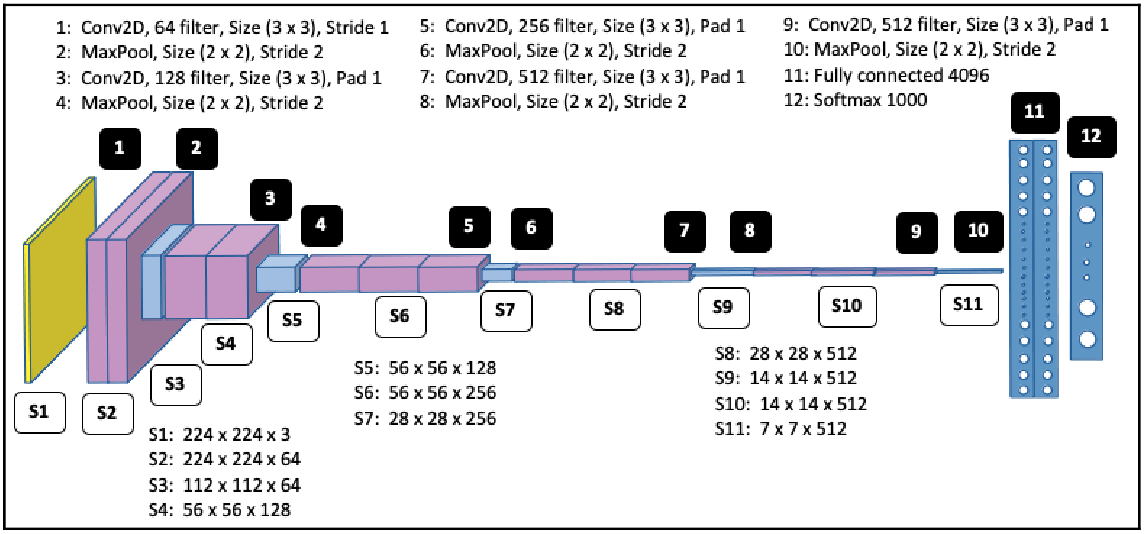

1
n/2
n
1
n/2
n
1
n/2
n

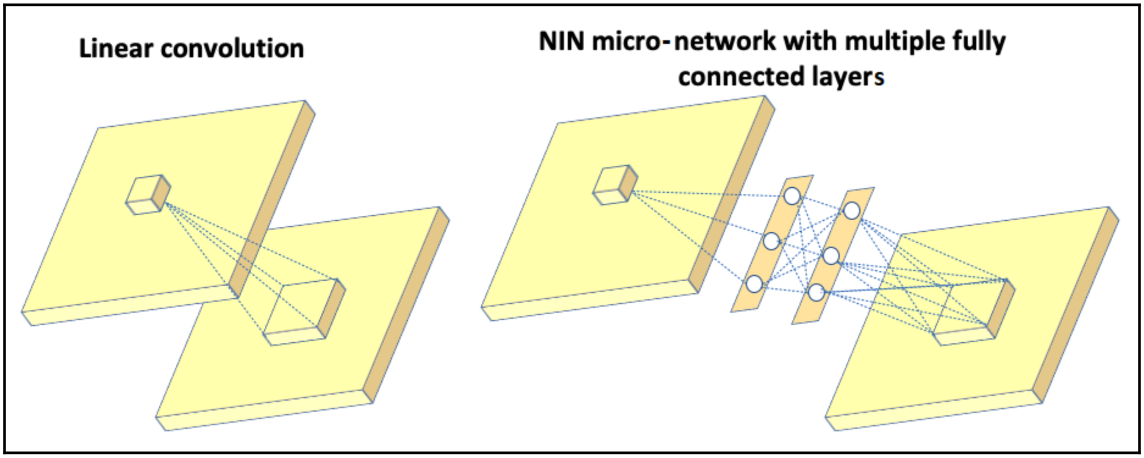

1 n/2 n. 1. n/2 n 1 n/2 n. 1. n/2 n 1 n/2 n. 1. n/2 n 1 n/2 n. 1. n/2 n. .1. n/2 n. 1. n/2

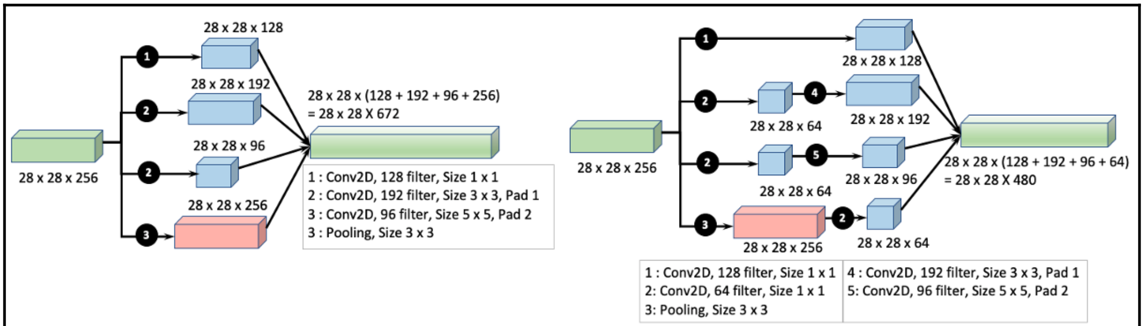

1
n/2
n

Chapter 5: Neural Network Architecture and Models

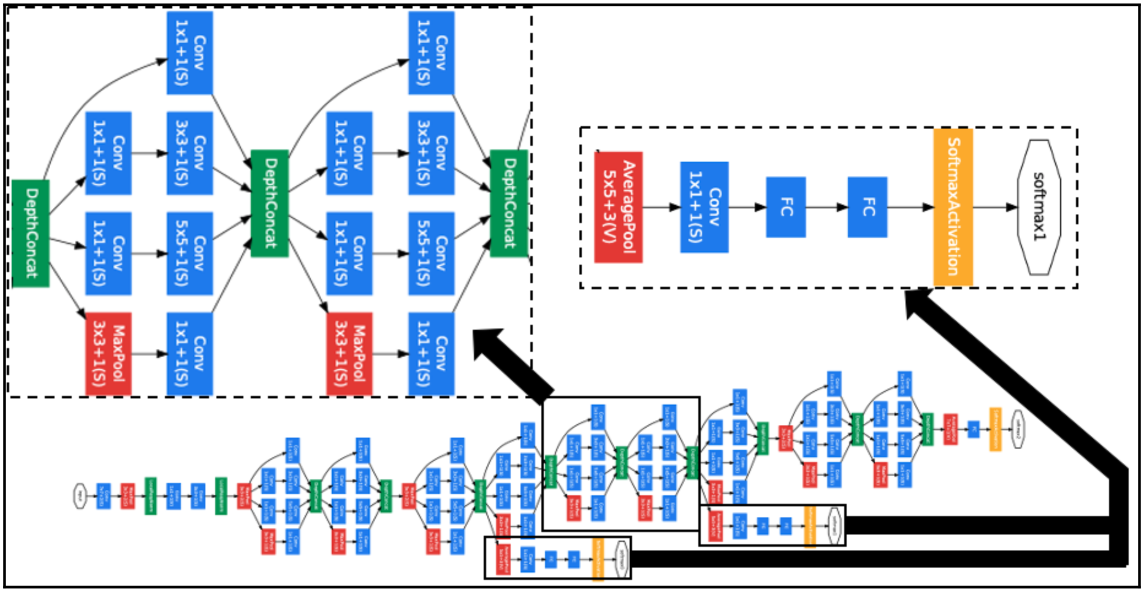


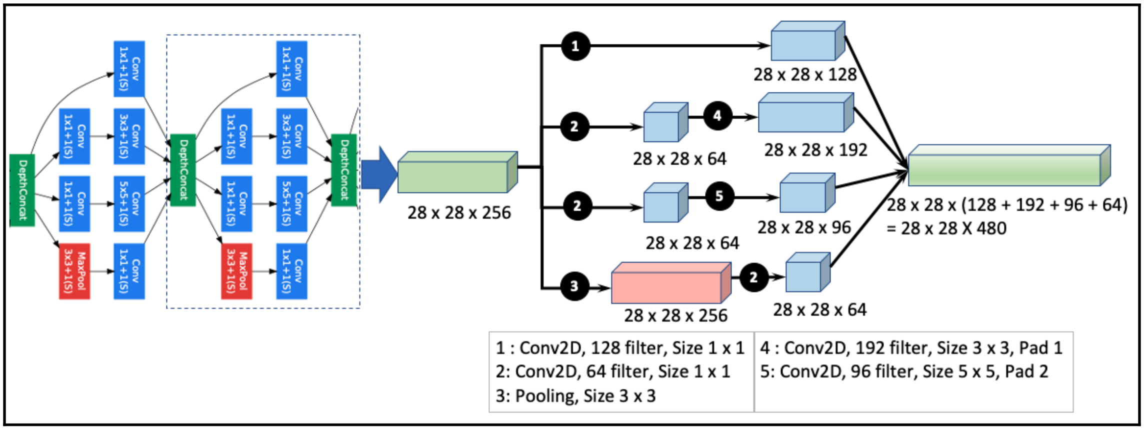


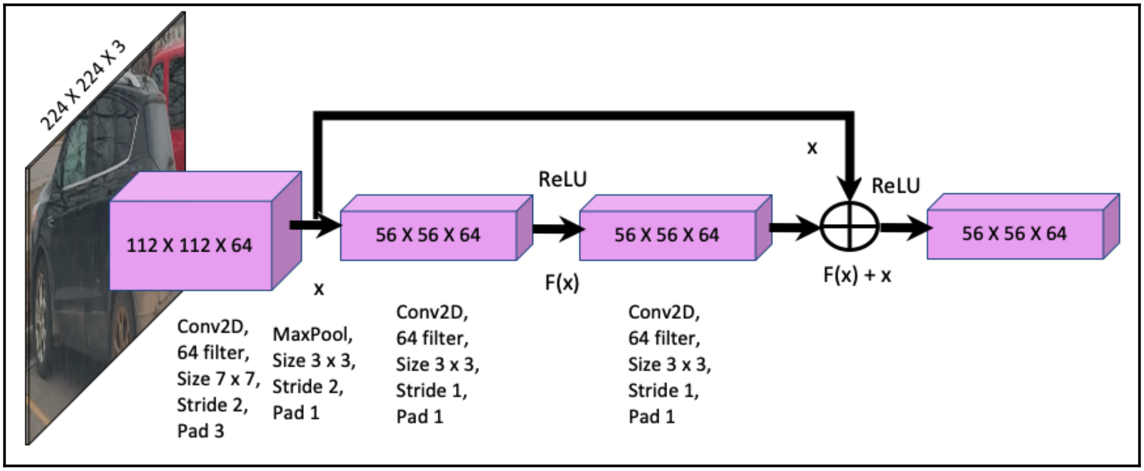


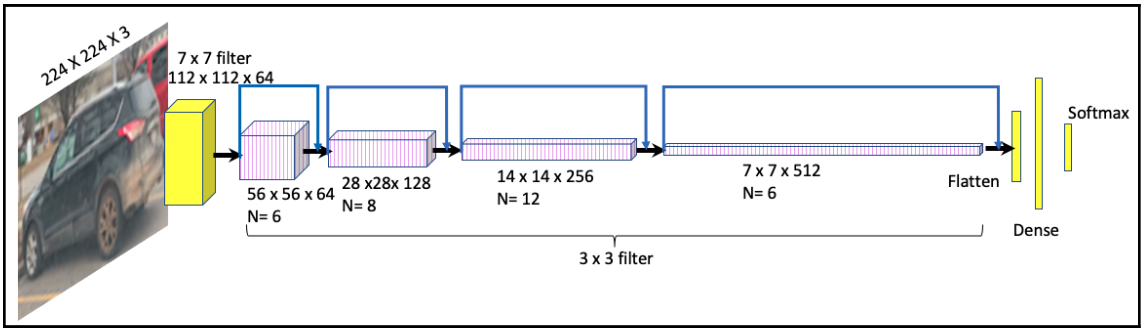


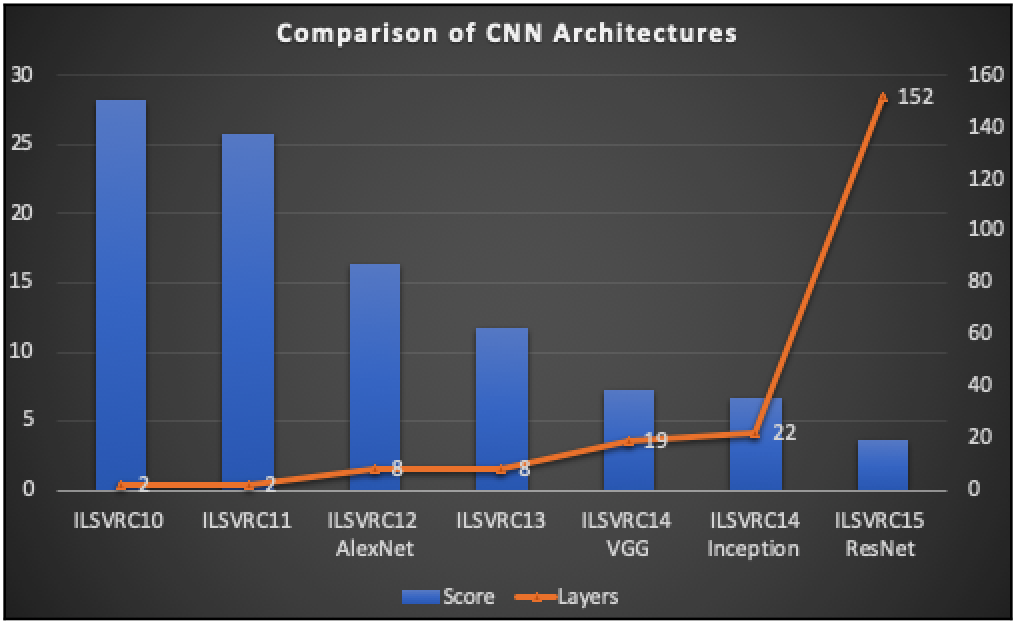
a) Inception layer without 1 x 1 leading to final dimensions of 28 x 28 x 672 **b) Inception layer with 1 x 1 leading to final dimensions of 28 x 28 x 480**


DepthConcat
Conv 1x1+1(S)
Conv 3x3+1(S)
Conv 5x5+1(S)
MaxPool 3x3+1(S)
AveragePool 5x5+3(V)
FC
SoftmaxActivation
softmax1

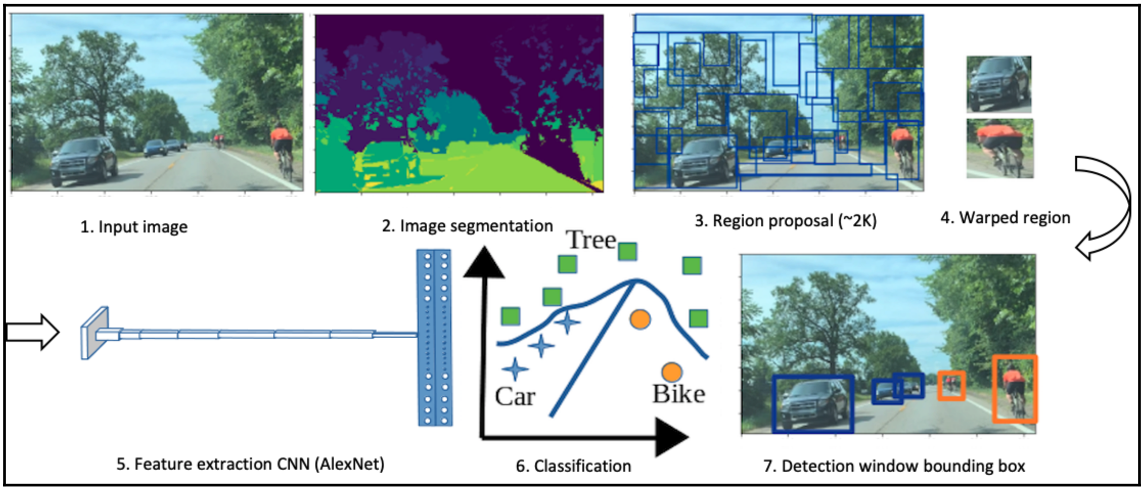

DepthConcat
Conv 1x1+1(S)
Conv 3x3+1(S)
Conv 5x5+1(S)
MaxPool 3x3+1(S)
28 x 28 x 256
28 x 28 x 128
28 x 28 x 64
28 x 28 x 192
28 x 28 x 64
28 x 28 x 96
28 x 28 x 256
28 x 28 x 64
28 x 28 x (128 + 192 + 96 + 64)
= 28 x 28 X 480
1 : Conv2D, 128 filter, Size 1 x 1
2: Conv2D, 64 filter, Size 1 x 1
3: Pooling, Size 3 x 3
4 : Conv2D, 192 filter, Size 3 x 3, Pad 1
5: Conv2D, 96 filter, Size 5 x 5, Pad 2


224 X 224 X 3
112 X 112 X 64
x
ReLU
56 X 56 X 64
56 X 56 X 64
x
ReLU
56 X 56 X 64
F(x)
F(x) + x
Conv2D,
64 filter,
Size 7 x 7,
Stride 2,
Pad 3
MaxPool,
Size 3 x 3,
Stride 2,
Pad 1
Conv2D,
64 filter,
Size 3 x 3,
Stride 1,
Pad 1
Conv2D,
64 filter,
Size 3 x 3,
Stride 1,
Pad 1

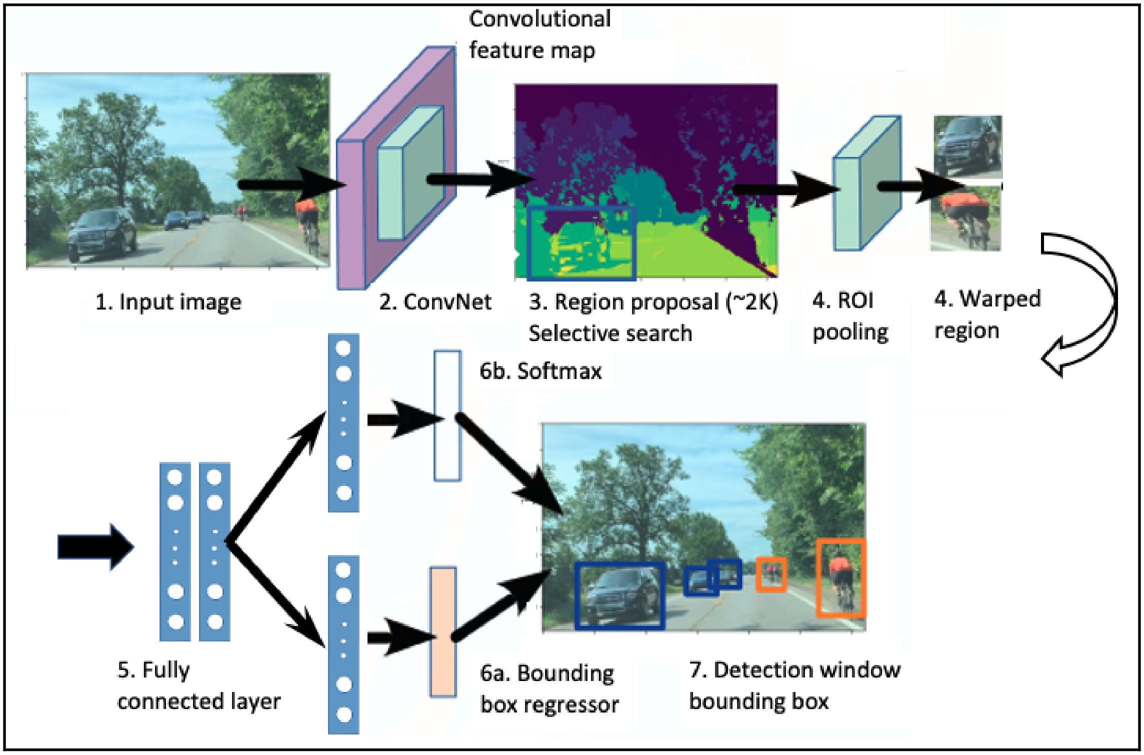

224 X 224 X 3
7 x 7 filter
112 x 112 x 64
56 x 56 x 64
N= 6
28 x28x 128
N= 8
14 x 14 x 256
N= 12
7 x 7 x 512
N= 6
Flatten
Softmax
Dense
3 x 3 filter

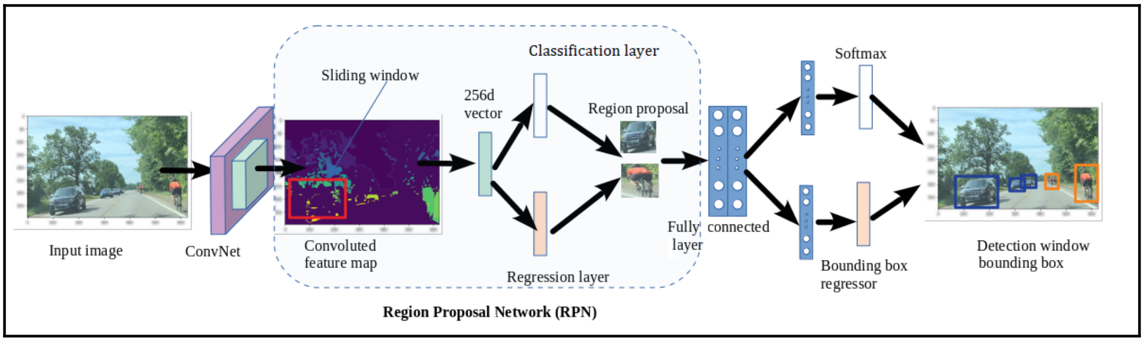


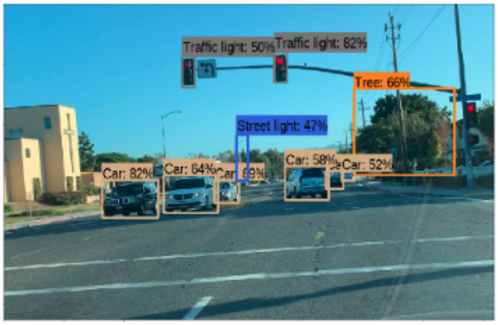
Comparison of CNN Architectures
30
25
20
15
10
5
0
160
140
120
100
80
60
40
20
0
152
2
2
8
8
19
22
ILSVRC10
ILSVRC11
ILSVRC12 AlexNet
ILSVRC13
ILSVRC14 VGG
ILSVRC14 Inception
ILSVRC15 ResNet
Score
Layers



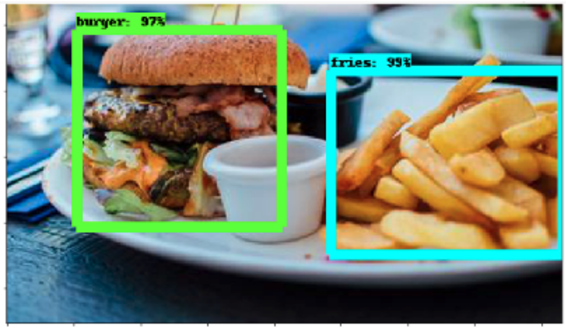
1. Input image
2. Image segmentation
3. Region proposal (~2K)
4. Warped region
Tree
Car
Bike
5. Feature extraction CNN (AlexNet)
6. Classification
7. Detection window bounding box


Convolutional
feature map
1. Input image
2. ConvNet
3. Region proposal (~2K)
Selective search
4. ROI
pooling
4. Warped
region
6b. Softmax
5. Fully
connected layer
6a. Bounding
box regressor
7. Detection window
bounding box

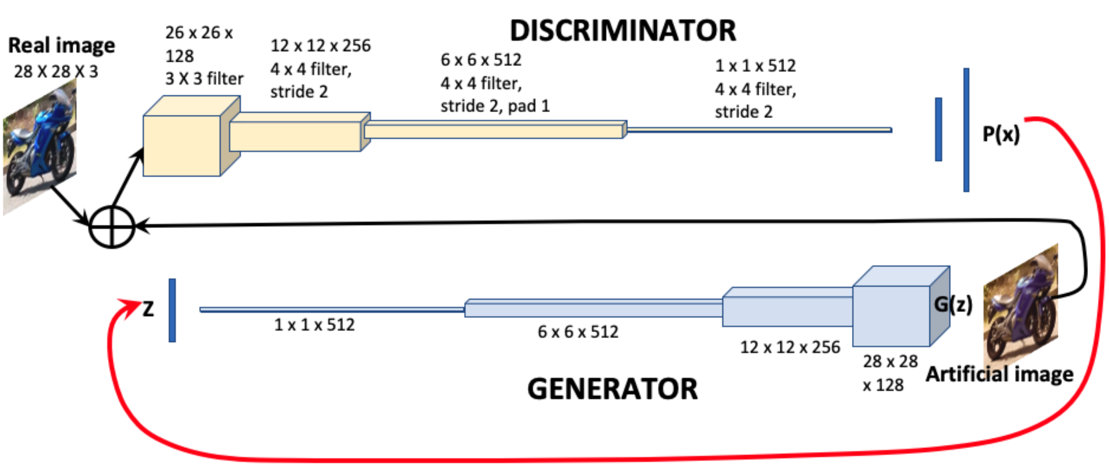

Classification layer
Sliding window
256d
vector
Region proposal
Softmax
Input image
ConvNet
Convoluted
feature map
Regression layer
Fully
layer
connected
Bounding box
regressor
Detection window
bounding box
Region Proposal Network (RPN)

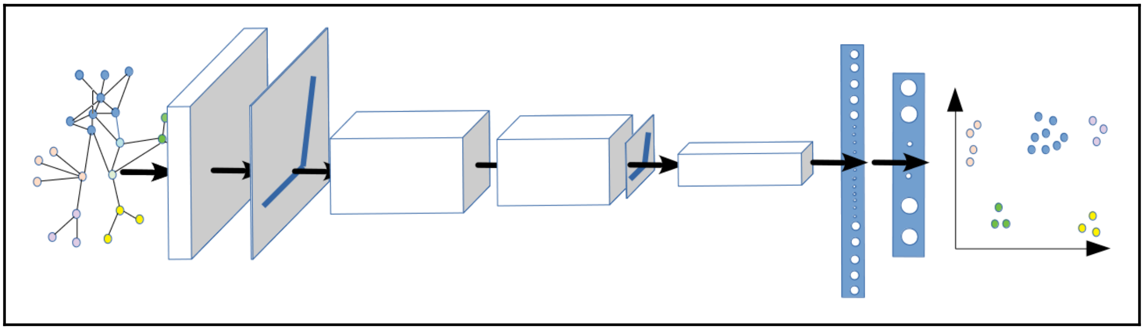


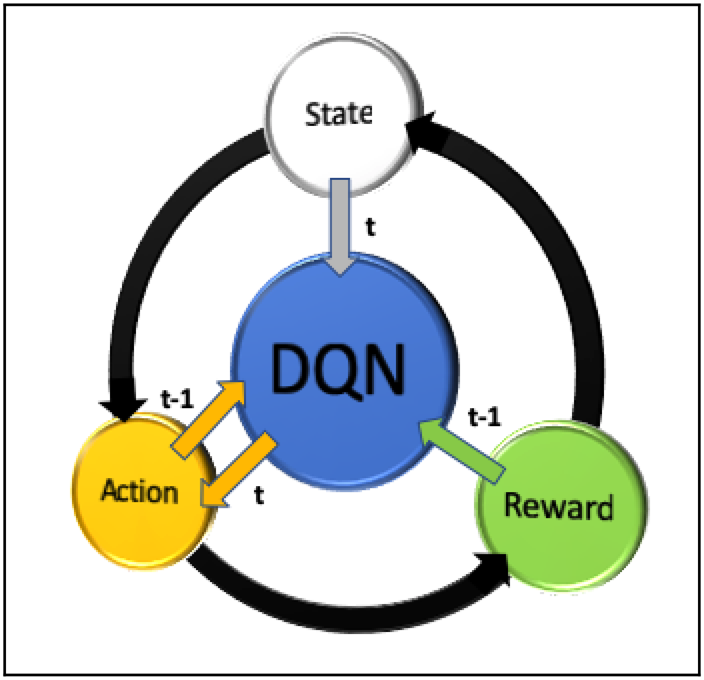
Faster R-CNN + Inception ResNet backend
- inference done using pretrained model from TensorFlow Hub



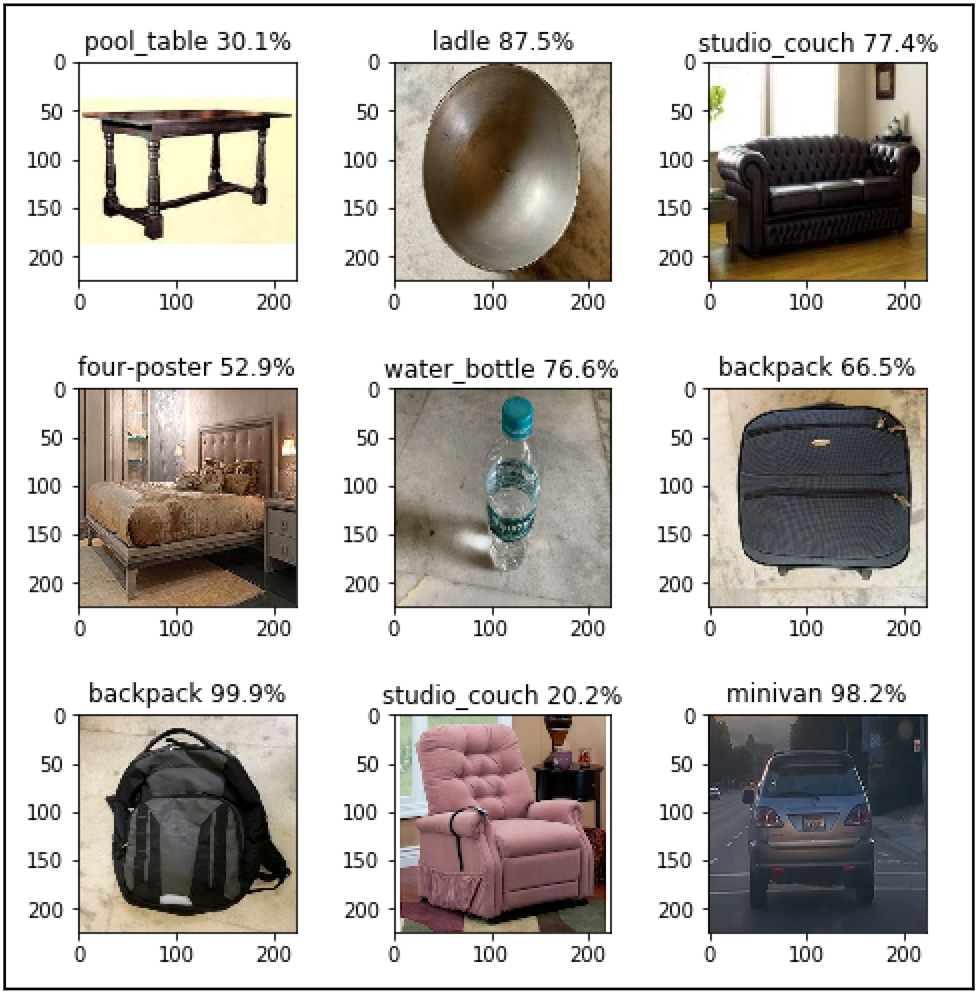
Faster R-CNN + Inception backend
- inference done by training custom model

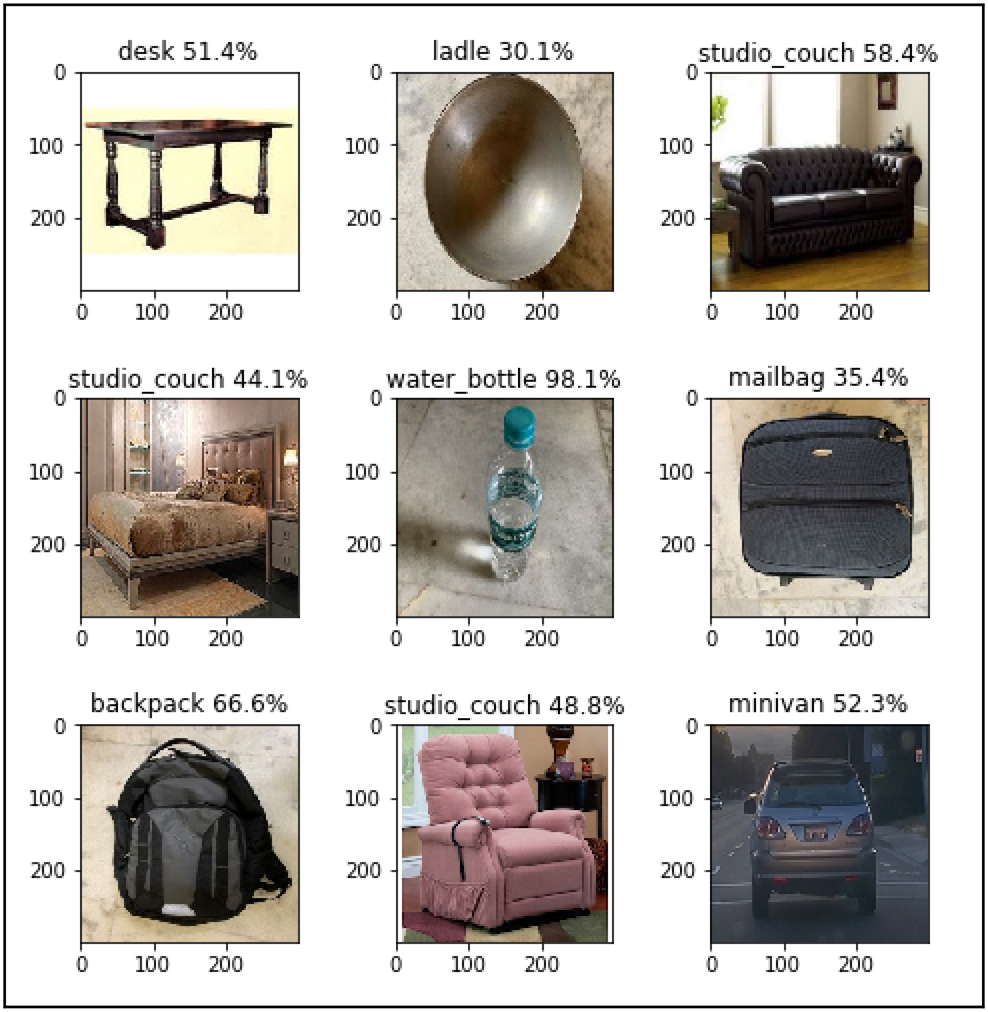





State
t
DQN
t-1
t-1
Action
t
Reward

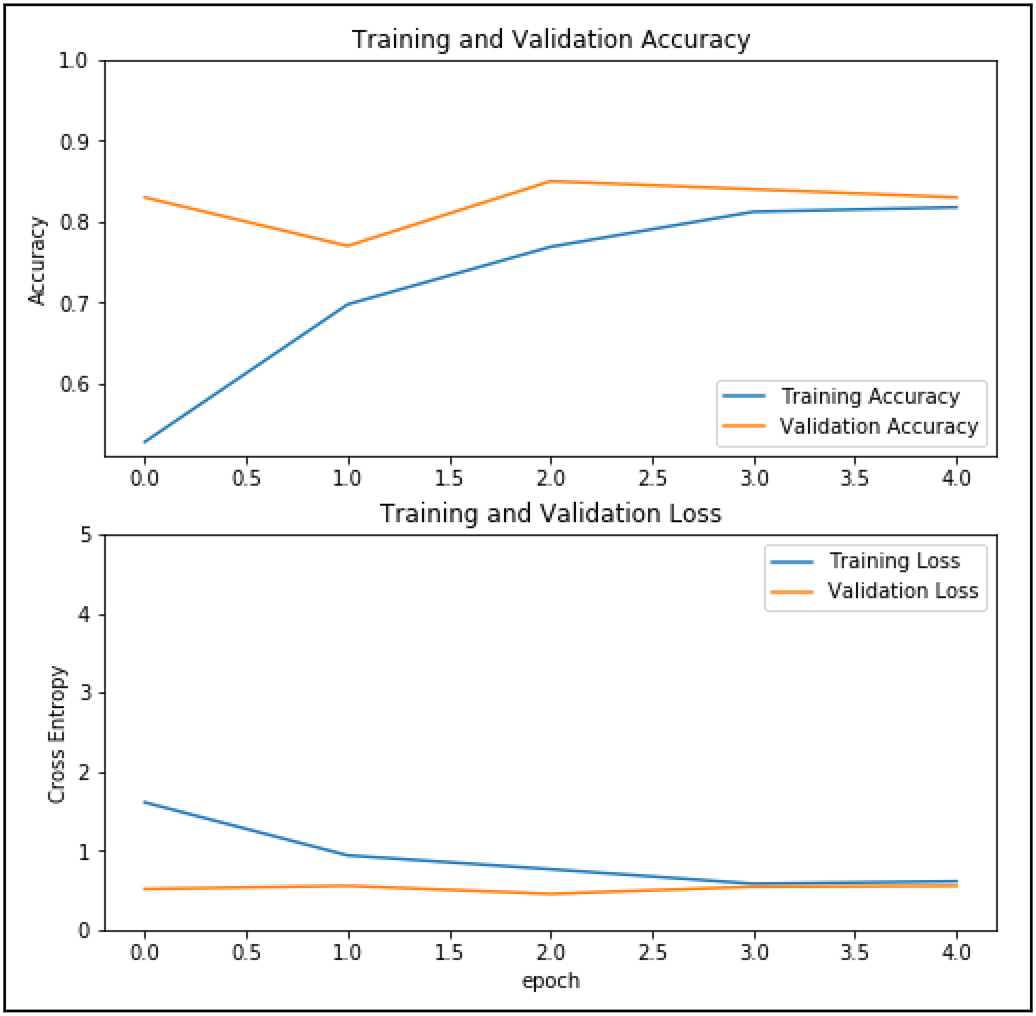
Chapter 6: Visual Search Using Transfer Learning


pool_table 30.1%
ladle 87.5%
studio_couch 77.4%
four-poster 52.9%
water_bottle 76.6%
backpack 66.5%
backpack 99.9%
studio_couch 20.2%
minivan 98.2%
0
50
100
150
200

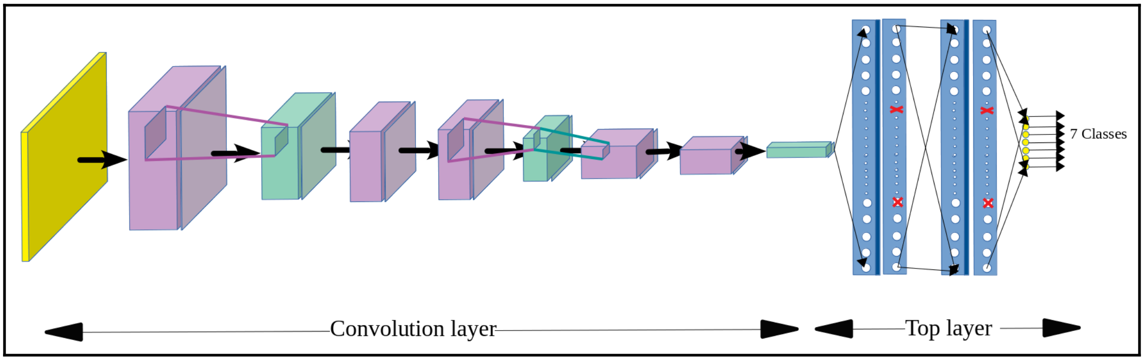

desk 51.4%
ladle 30.1%
studio_couch 58.4%
studio_couch 44.1%
water_bottle 98.1%
mailbag 35.4%
backpack 66.6%
studio_couch 48.8%
minivan 52.3%

Training and Validation Accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.4
Accuracy
Training Accuracy
Validation Accuracy
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Training and Validation Loss
5
4
3
2
1
0
Cross Entropy
Training Loss
Validation Loss
epoch

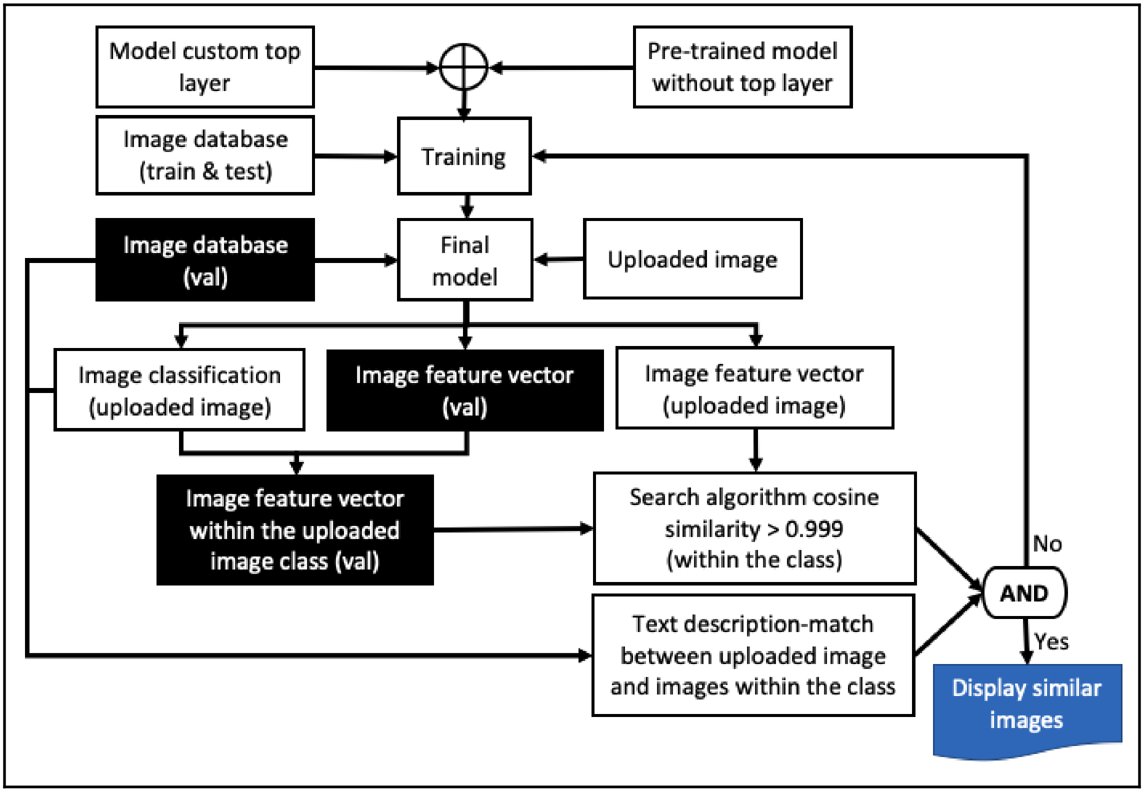
Training and Validation Accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.4
Accuracy
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Training Accuracy
Validation Accuracy
Training and Validation Loss
5
4
3
2
1
0
Cross Entropy
Training Loss
Validation Loss
epoch

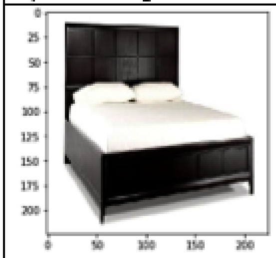
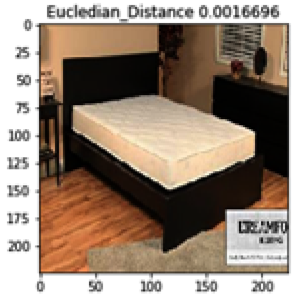


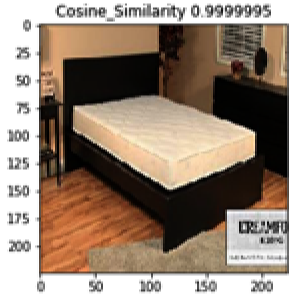
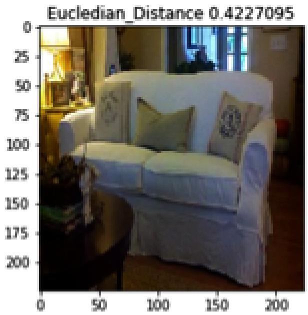
Training and Validation Accuracy
Accuracy
1.0
0.9
0.8
0.7
0.6
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Training Accuracy
Validation Accuracy
Training and Validation Loss
Cross Entropy
5
4
3
2
1
0
Training Loss
Validation Loss
epoch

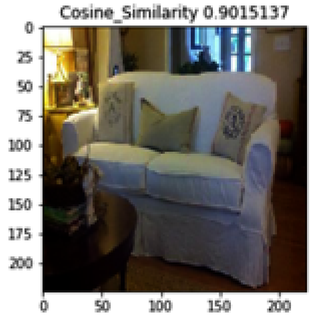


7 Classes
Convolution layer
Top layer


Model custom top layer
Pre-trained model without top layer
Image database (train & test)
Training
Image database (val)
Final model
Uploaded image
Image classification (uploaded image)
Image feature vector (val)
Image feature vector (uploaded image)
Image feature vector within the uploaded image class (val)
Search algorithm cosine similarity > 0.999 (within the class)
No
AND
Yes
Text description-match between uploaded image and images within the class
Display similar images

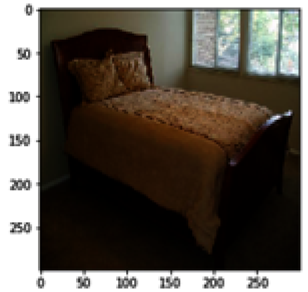
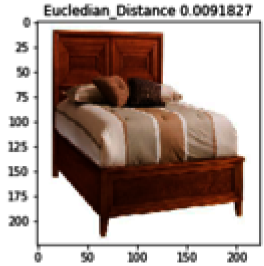
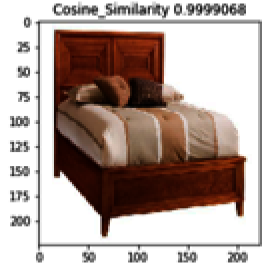
<table>
<tr><th>Uploaded image</th><th>Display image</th></tr>
<tr><td rowspan="3"></td><td>ResNet – correct prediction


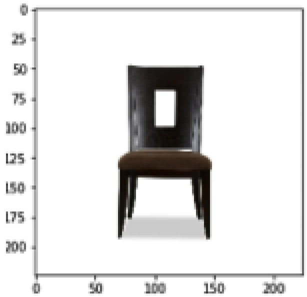
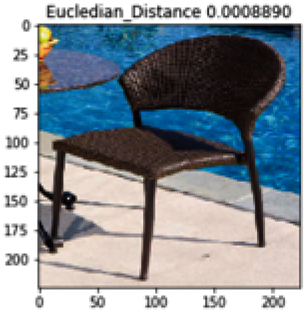

</td></tr>
<tr><td>Inception – incorrect class prediction


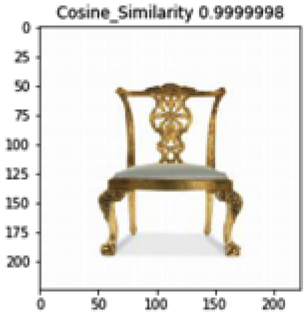
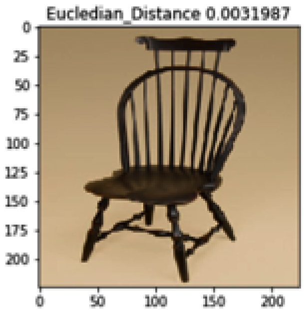

</td></tr>
<tr><td>VGG – Correct prediction

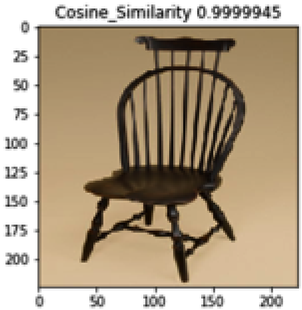
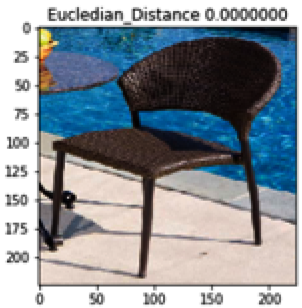


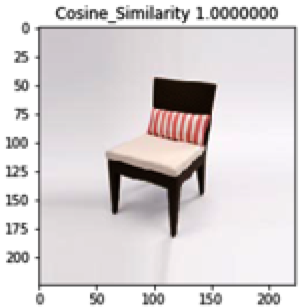
</td></tr>
</table>

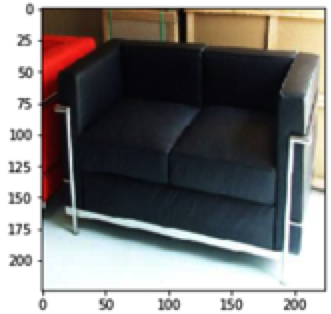
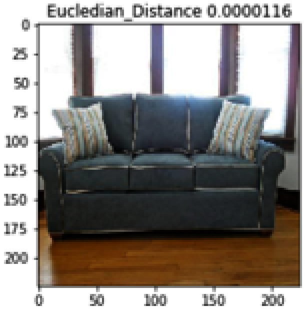
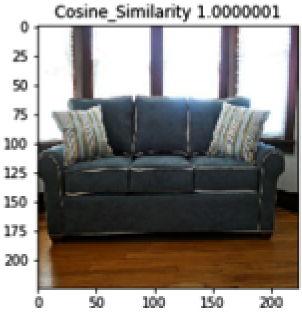
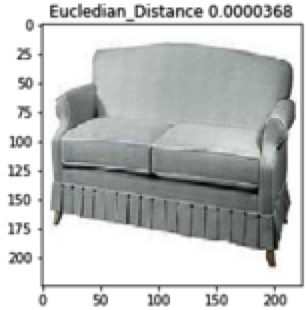
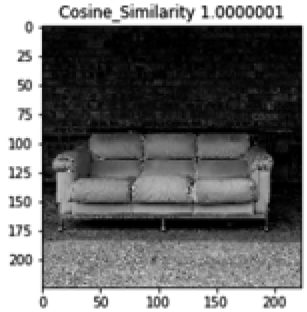
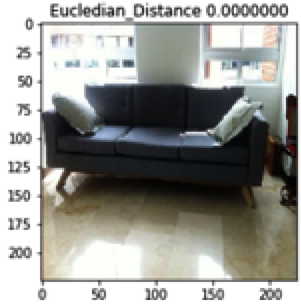
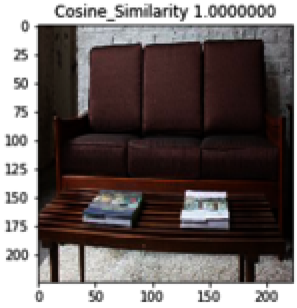
Uploaded image	Display image (correct prediction) – Inception – bed class
	 

<table>
<tr><th>Uploaded image</th><th>Display image</th></tr>
<tr><td rowspan="3"></td><td>ResNet – correct prediction


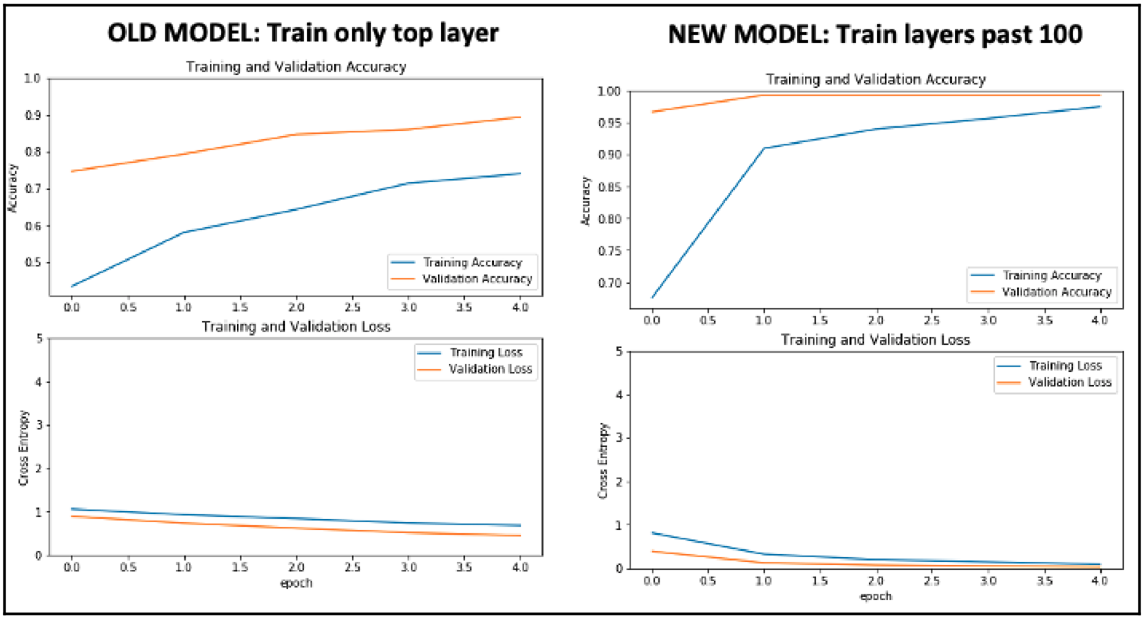

</td></tr>
<tr><td>Inception – Correct prediction



</td></tr>
<tr><td>VGG – Correct prediction


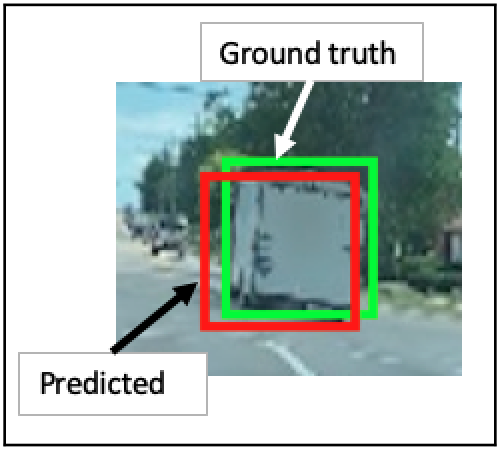

</td></tr>
</table>

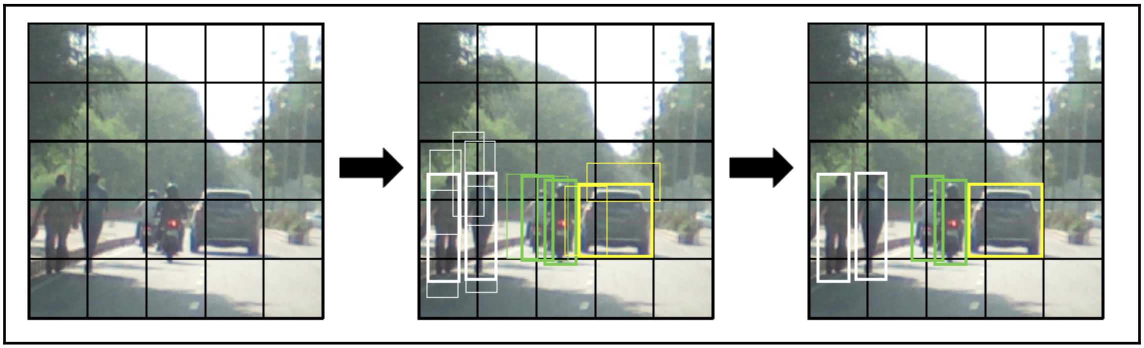
Uploaded image	Display image
	ResNet – correct prediction  
	Inception – Correct prediction  
	VGG – Correct prediction  

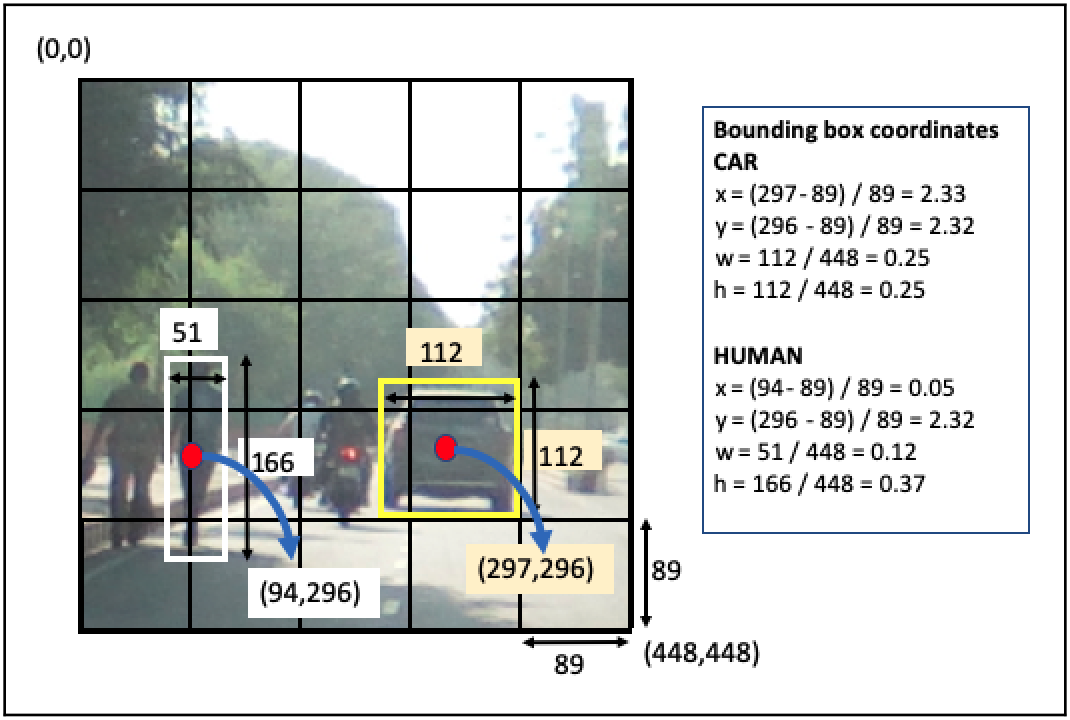
```
timeit(train_generator)

.....................................................
1000 batches: 142.58274674415588 s
70.13471 Images/s

timeit(train_ds)

.....................................................
1000 batches: 1.5855846405029297 s
6306.82194 Images/s
```



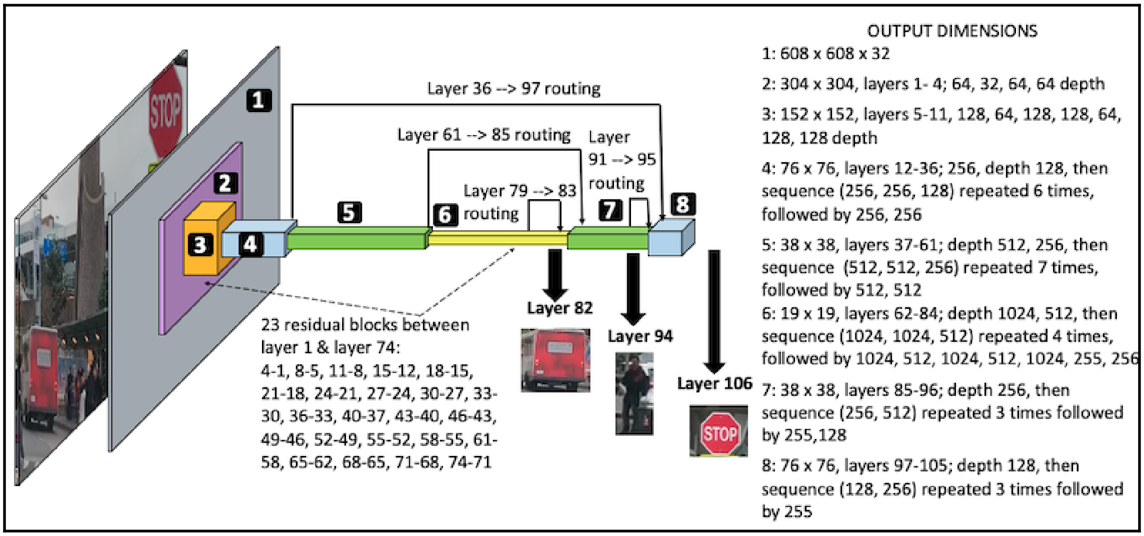
Chapter 7: Object Detection Using YOLO



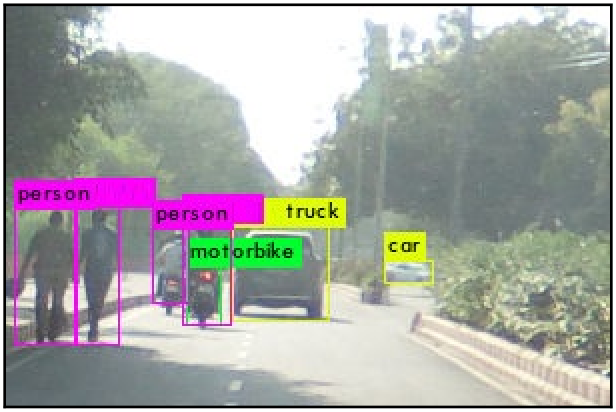


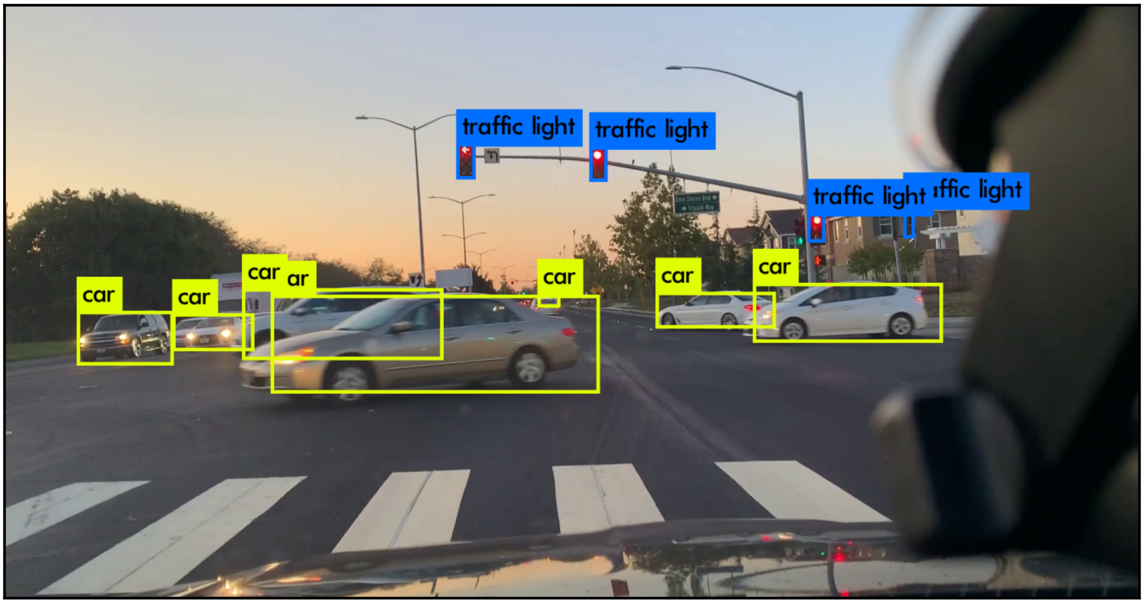


(0,0)
51
112
166
112
(94,296)
(297,296)
89
89
(448,448)
Bounding box coordinates
CAR
x = (297 - 89) / 89 = 2.33
y = (296 - 89) / 89 = 2.32
w = 112 / 448 = 0.25
h = 112 / 448 = 0.25
HUMAN
x = (94 - 89) / 89 = 0.05
y = (296 - 89) / 89 = 2.32
w = 51 / 448 = 0.12
h = 166 / 448 = 0.37



1
2
3
4
5
6
7
8
Layer 36 --> 97 routing
Layer 61 --> 85 routing
Layer 91 --> 95 routing
Layer 79 --> 83 routing
Layer 82
Layer 94
Layer 106
23 residual blocks between layer 1 & layer 74:
4-1, 8-5, 11-8, 15-12, 18-15, 21-18, 24-21, 27-24, 30-27, 33-30, 36-33, 40-37, 43-40, 46-43, 49-46, 52-49, 55-52, 58-55, 61-58, 65-62, 68-65, 71-68, 74-71
OUTPUT DIMENSIONS
1: 608 x 608 x 32
2: 304 x 304, layers 1- 4; 64, 32, 64, 64 depth
3: 152 x 152, layers 5-11, 128, 64, 128, 128, 64, 128, 128 depth
4: 76 x 76, layers 12-36; 256, depth 128, then sequence (256, 256, 128) repeated 6 times, followed by 256, 256
5: 38 x 38, layers 37-61; depth 512, 256, then sequence (512, 512, 256) repeated 7 times, followed by 512, 512
6: 19 x 19, layers 62-84; depth 1024, 512, then sequence (1024, 1024, 512) repeated 4 times, followed by 1024, 512, 1024, 512, 1024, 255, 256
7: 38 x 38, layers 85-96; depth 256, then sequence (256, 512) repeated 3 times followed by 255,128
8: 76 x 76, layers 97-105; depth 128, then sequence (128, 256) repeated 3 times followed by 255


person
person
truck
motorbike
car

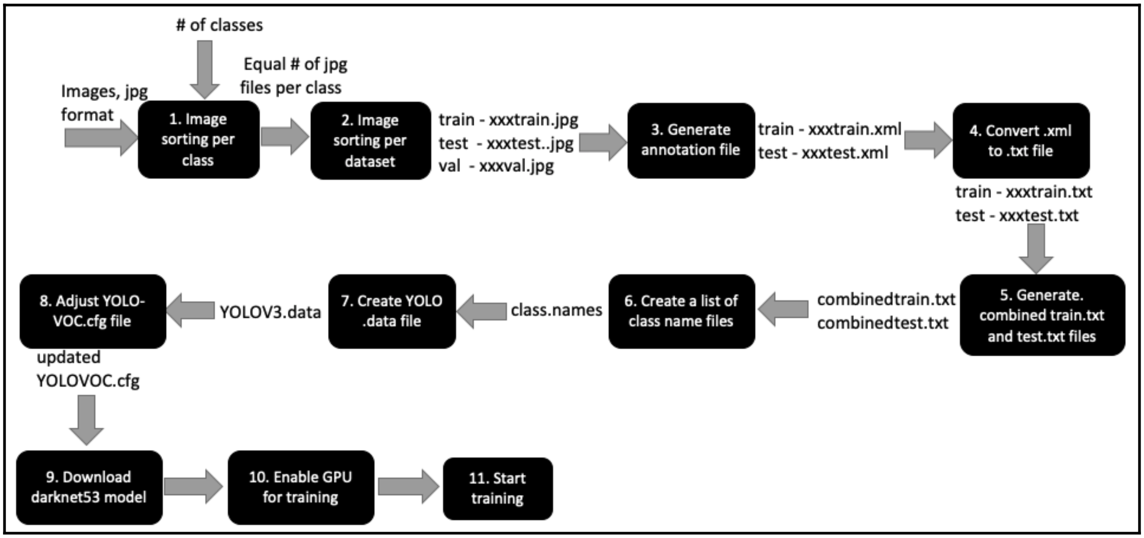

traffic light
traffic light
traffic light
car
car
car
car
car
car

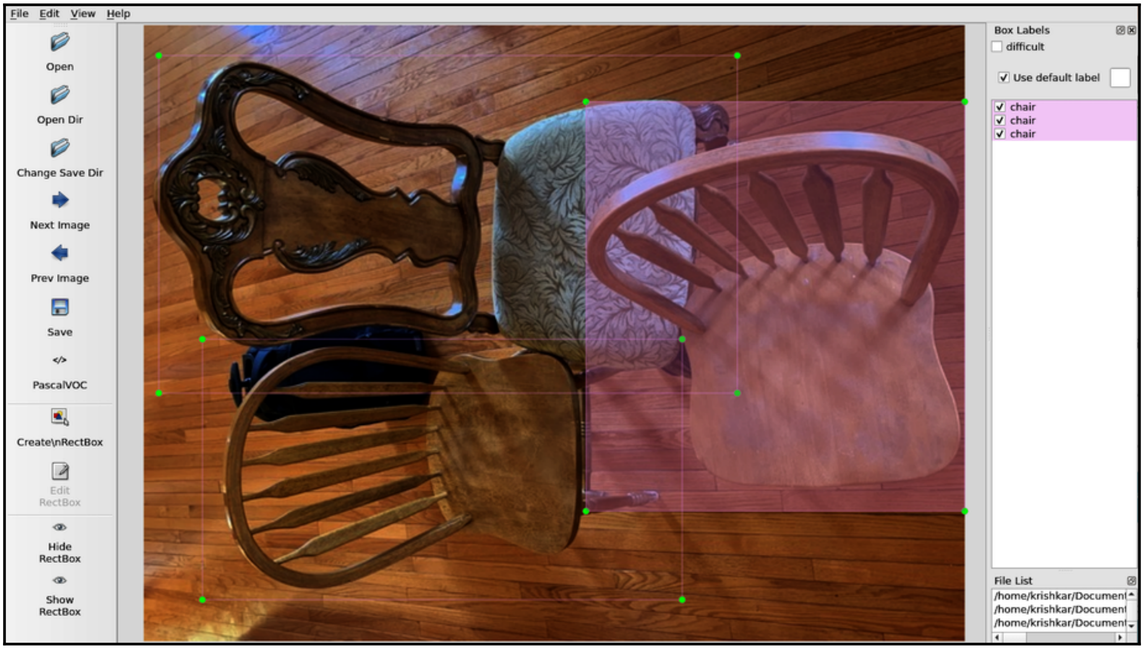
YOLOV2

Type	Filters	Size	Output
Convolutional	32	3 x 3	224 x 224
Maxpool		2 x 2 / 2	112 x 112
Convolutional	64	3 x 3	112 x 112
Maxpool		2 x 2 / 2	56 x 56
Convolutional	128	3 x 3	56 x 56
Convolutional	64	1 x 1	56 x 56
Convolutional	128	3 X 3	56 X 56
Maxpool		2 x 2 / 2	28 X 28
Convolutional	256	3 x 3	28 X 28
Convolutional	128	1 x 1	28 X 28
Convolutional	256	3 X 3	28 X 28
Maxpool		2 x 2 / 2	14 X 14
Convolutional	512	3 x 3	14 X 14
Convolutional	256	1 x 1	14 X 14
Convolutional	512	3 X 3	14 X 14
		2 x 2 / 2	7 X 7
Convolutional	1024	3 x 3	7 X 7
Convolutional	512	1 x 1	7 X 7
Convolutional	1024	3 X 3	7 X 7
Convolutional	512	1 x 1	7 X 7
Convolutional	1024	3 X 3	7 X 7
Convolutional	1000	1 X 1	7 X 7
Avgpool		Global	1000

YOLOV3

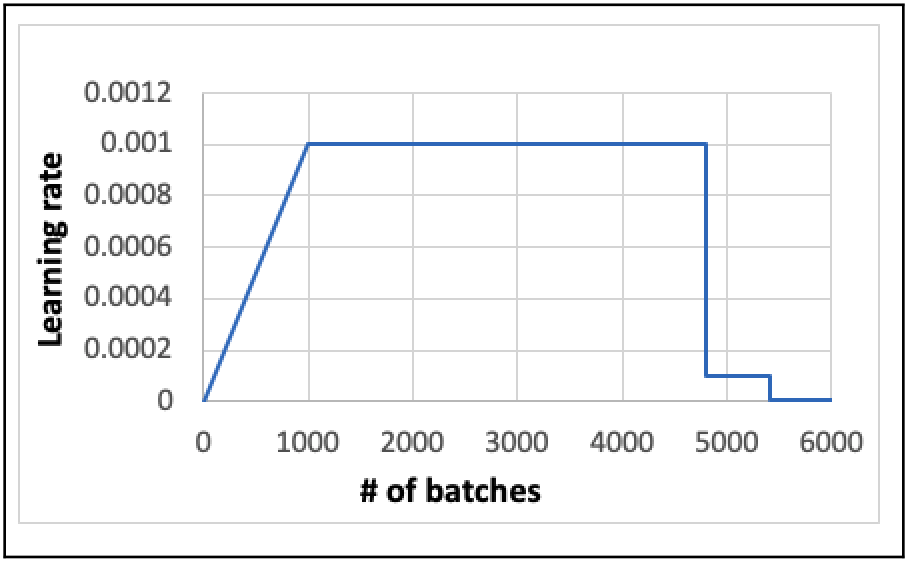
	Type	Filters	Size	Output
	Convolutional	32	3 x 3	256 x 256
	Convolutional	64	3 x 3 / 2	128 x 128
	Convolutional	32	1 x 1	
1 x	Convolutional	64	3 X 3	
	Residual			128 X 128
	Convolutional	128	3 x 3 / 2	64 x 64
	Convolutional	64	1 x 1	
2 x	Convolutional	128	3 x 3	
	Residual			64 X 64
	Convolutional	256	3 x 3 / 2	32 x 32
	Convolutional	128	1 x 1	
8 x	Convolutional	256	3 x 3	
	Residual			32 x 32
	Convolutional	512	3 x 3 / 2	32 x 32
	Convolutional	256	1 x 1	
8 x	Convolutional	512	3 x 3	
	Residual			16 x 16
	Convolutional	1024	3 x 3 / 2	32 x 32
	Convolutional	512	1 x 1	
4 x	Convolutional	1024	3 x 3	
	Residual			8 x 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

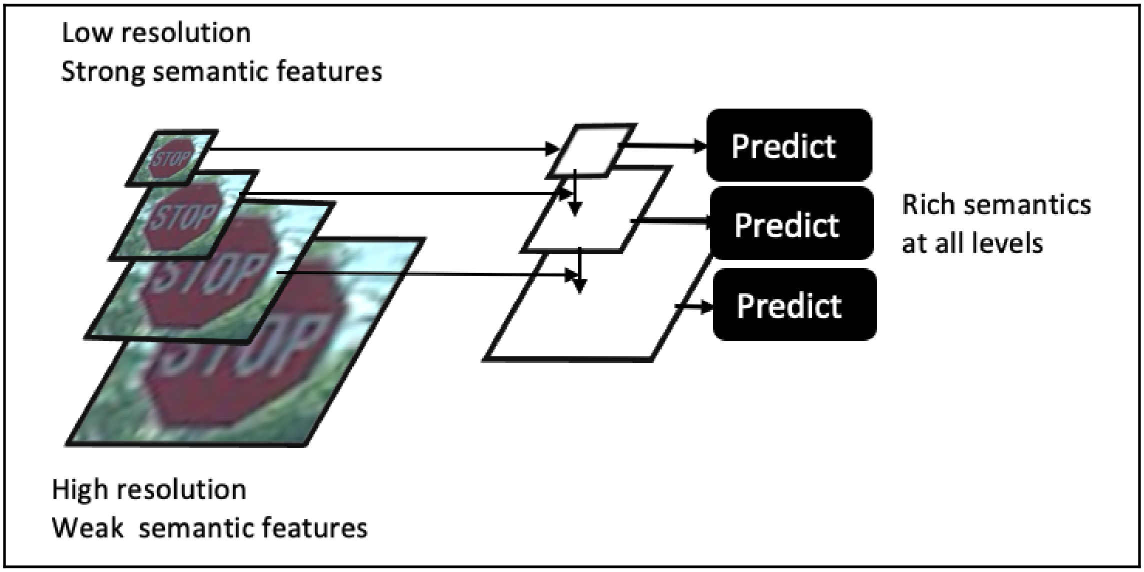




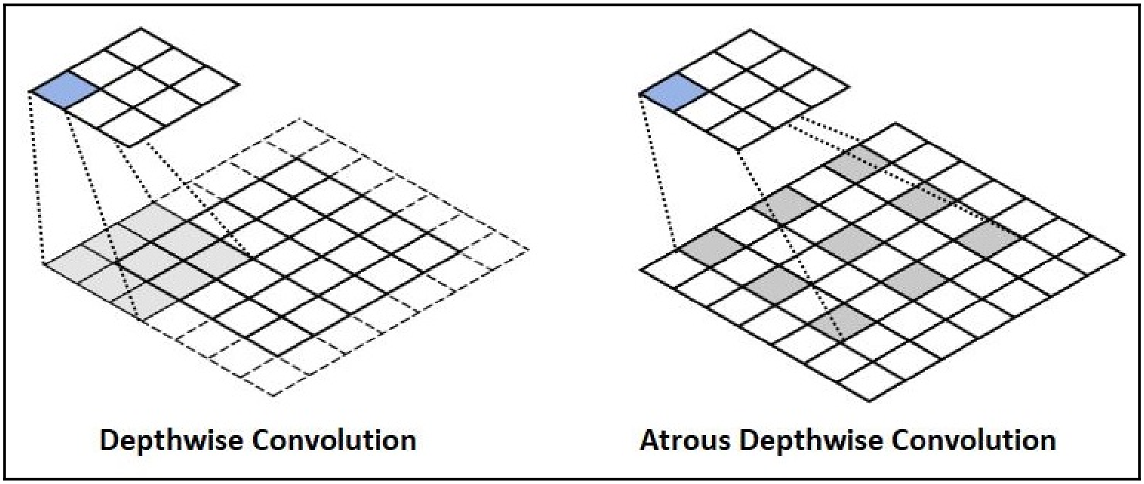
train.txt

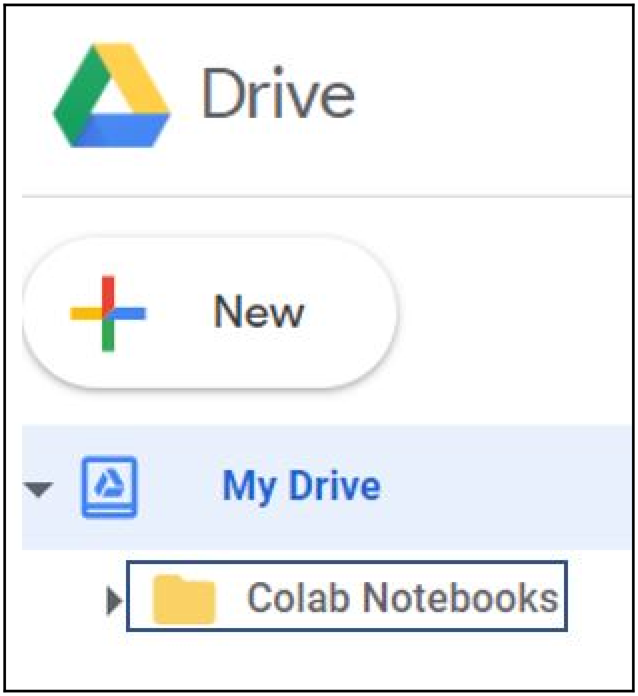
```
/home/krishkar/Documents/chapter7_yolo/furniture_data/trainyolo/sofa_316.jpg
/home/krishkar/Documents/chapter7_yolo/furniture_data/trainyolo/bed_418.jpg
/home/krishkar/Documents/chapter7_yolo/furniture_data/trainyolo/chair_169.jpg
/home/krishkar/Documents/chapter7_yolo/furniture_data/trainyolo/sofa_102.jpg
/home/krishkar/Documents/chapter7_yolo/furniture_data/trainyolo/chair_227.jpg
/home/krishkar/Documents/chapter7_yolo/furniture_data/trainyolo/bed_144.jpg
/home/krishkar/Documents/chapter7_yolo/furniture_data/trainyolo/chair_312.jpg
```

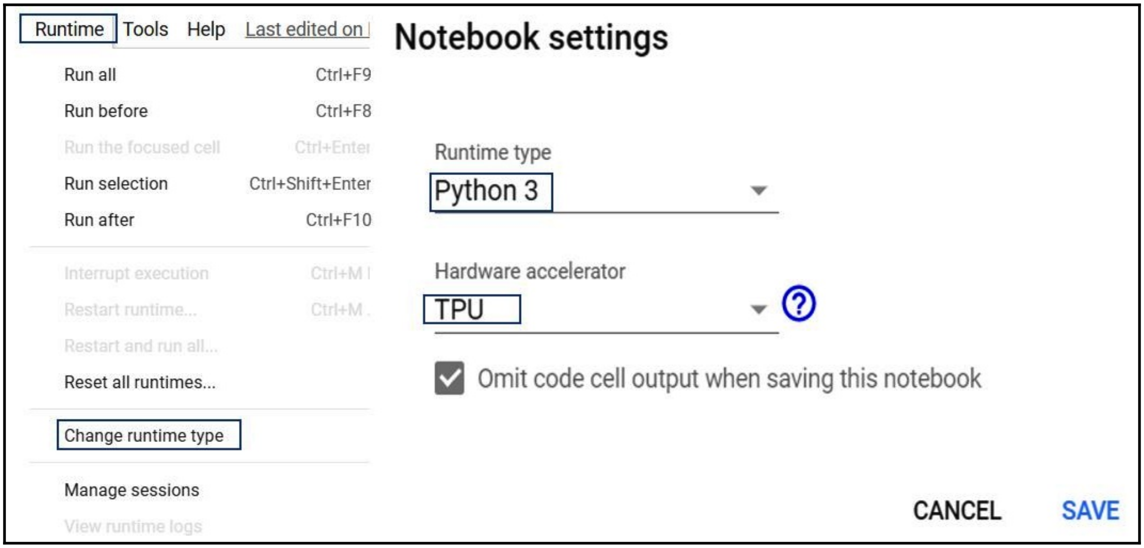

0.0012
0.001
0.0008
0.0006
0.0004
0.0002
0
Learning rate
0
1000
2000
3000
4000
5000
6000
# of batches

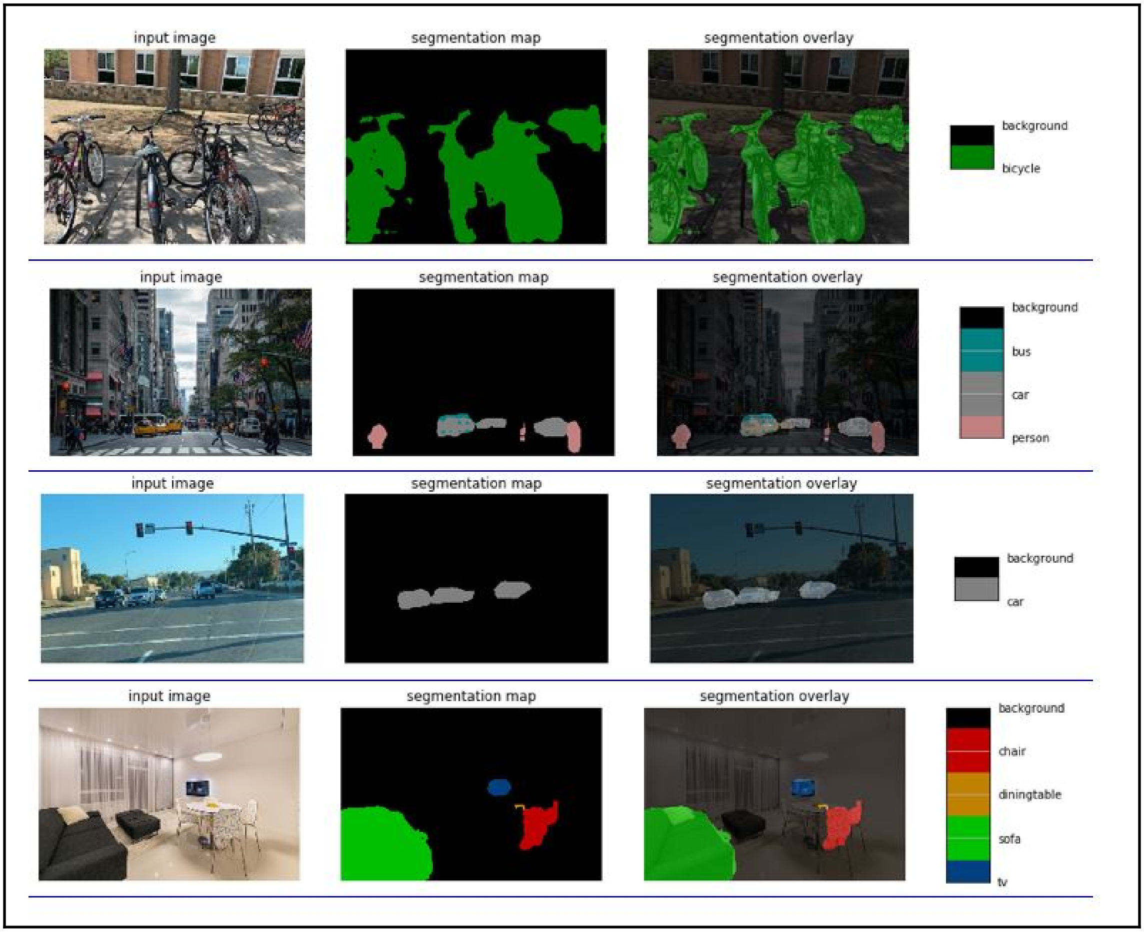

Low resolution
Strong semantic features
STOP
Predict
Predict
Predict
Rich semantics
at all levels
High resolution
Weak semantic features

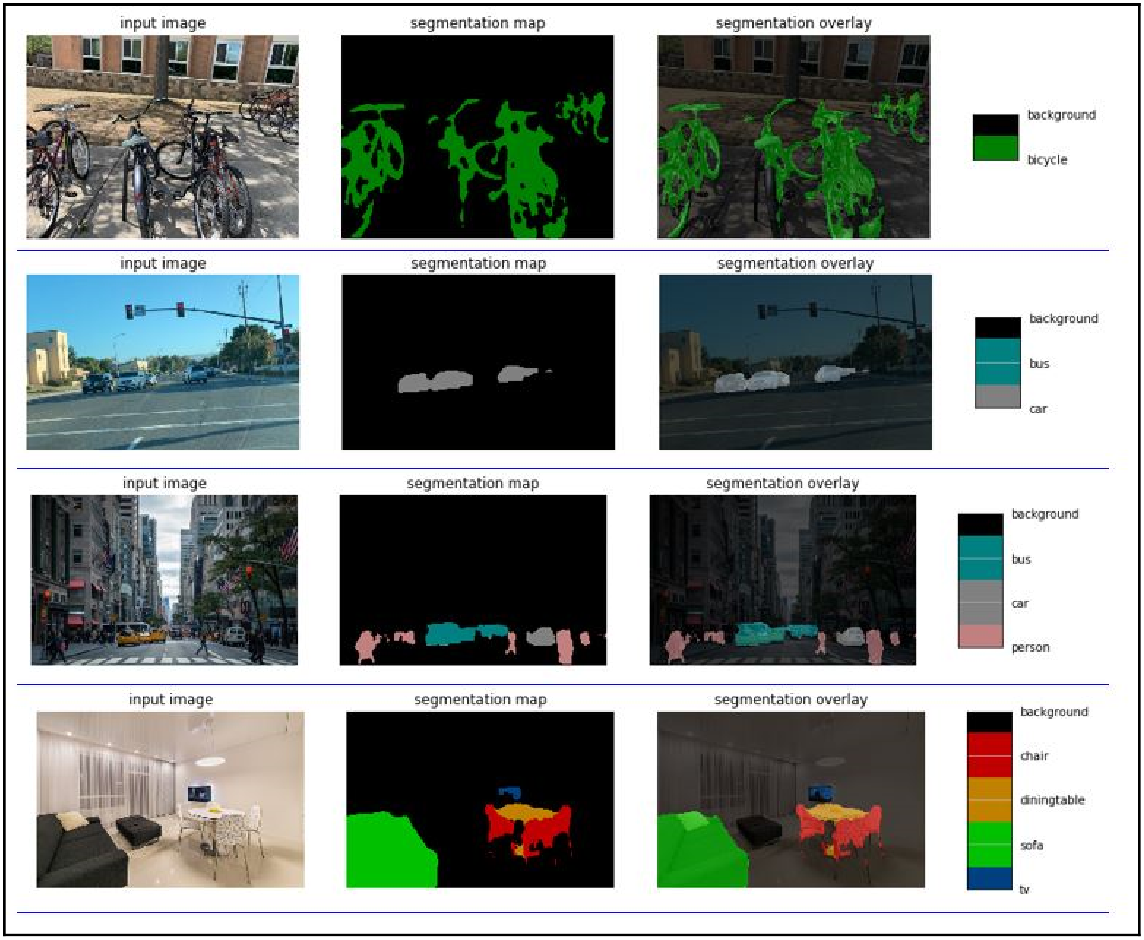
Chapter 8: Semantic Segmentation and Neural Style Transfer

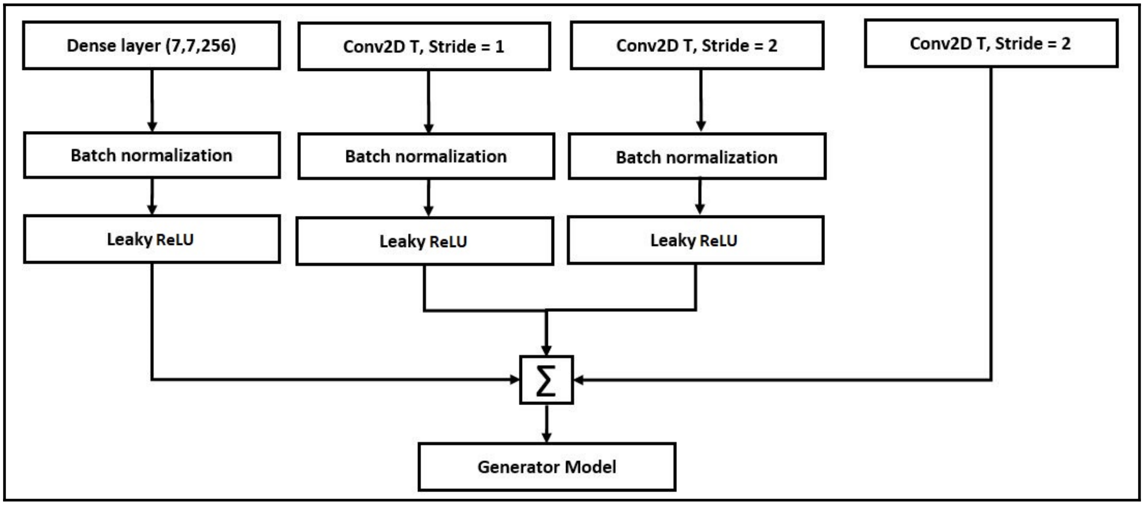


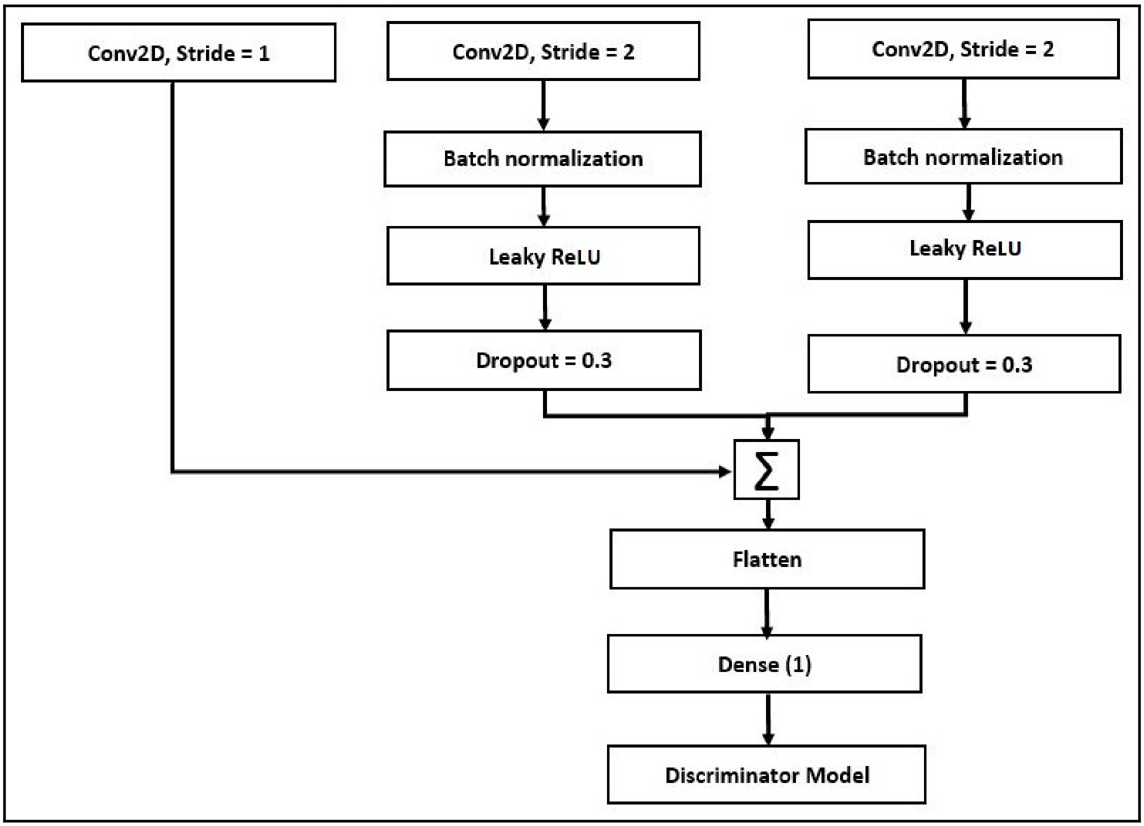


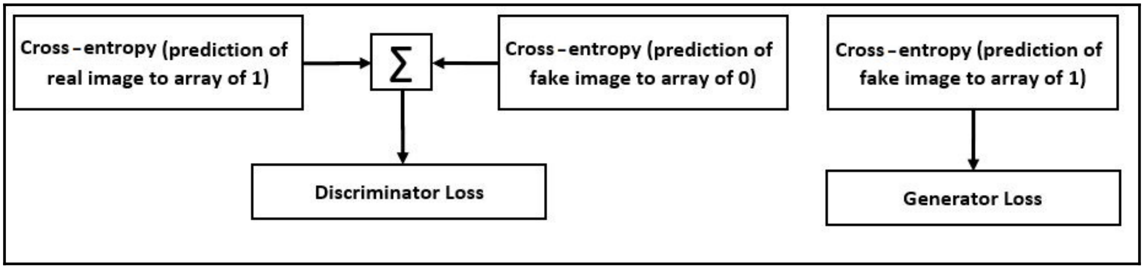

Runtime
Tools
Help
Last edited on
Run all
Ctrl+F9
Run before
Ctrl+F8
Run the focused cell
Ctrl+Enter
Run selection
Ctrl+Shift+Enter
Run after
Ctrl+F10
Interrupt execution
Restart runtime...
Restart and run all...
Reset all runtimes...
Change runtime type
Manage sessions
View runtime logs
Notebook settings
Runtime type
Python 3
Hardware accelerator
TPU
Omit code cell output when saving this notebook
CANCEL
SAVE

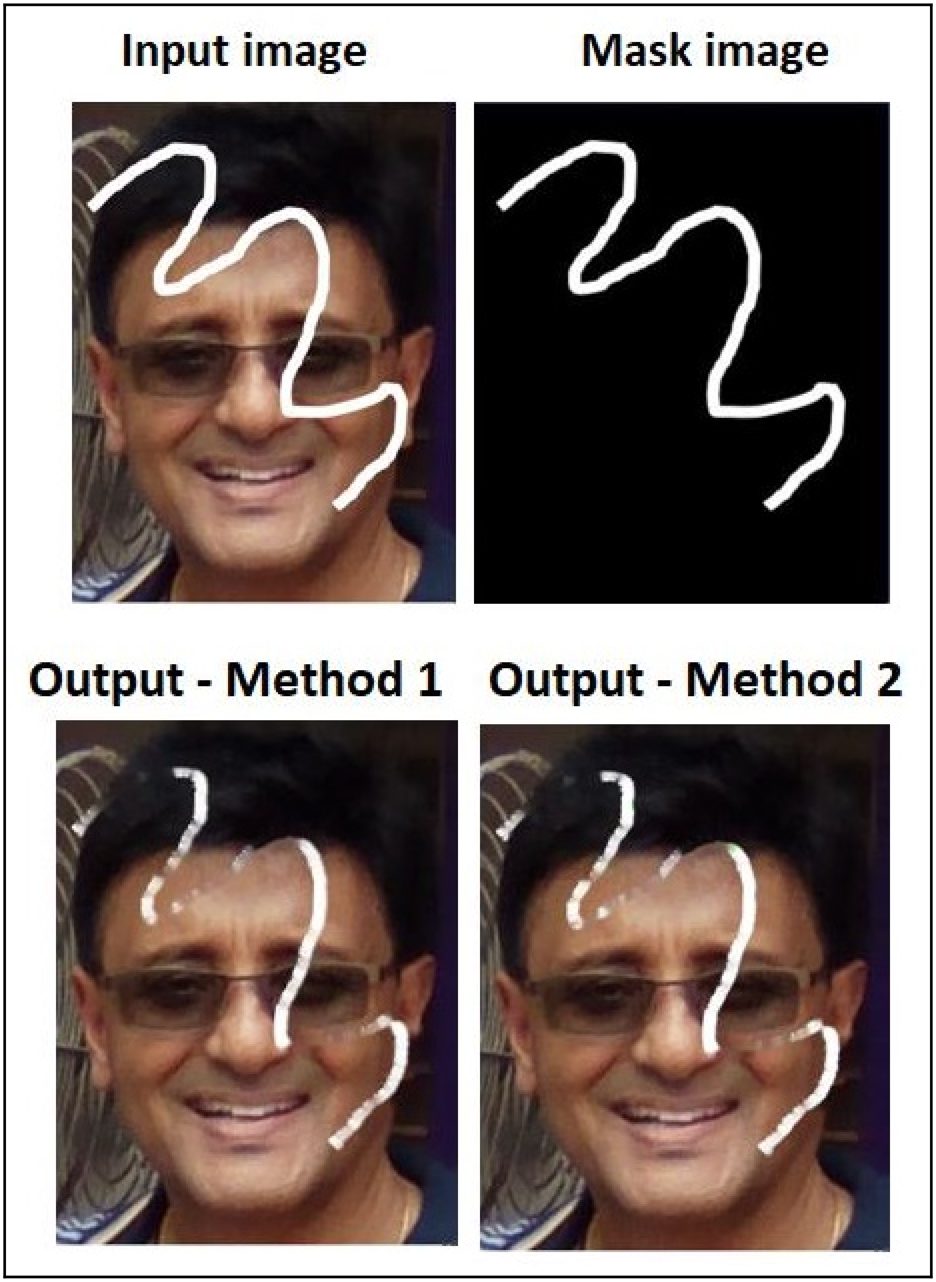

input image
segmentation map
segmentation overlay
background
bicycle
input image
segmentation map
segmentation overlay
background
bus
car
person
input image
segmentation map
segmentation overlay
background
car
input image
segmentation map
segmentation overlay
background
chair
diningtable
sofa
tv

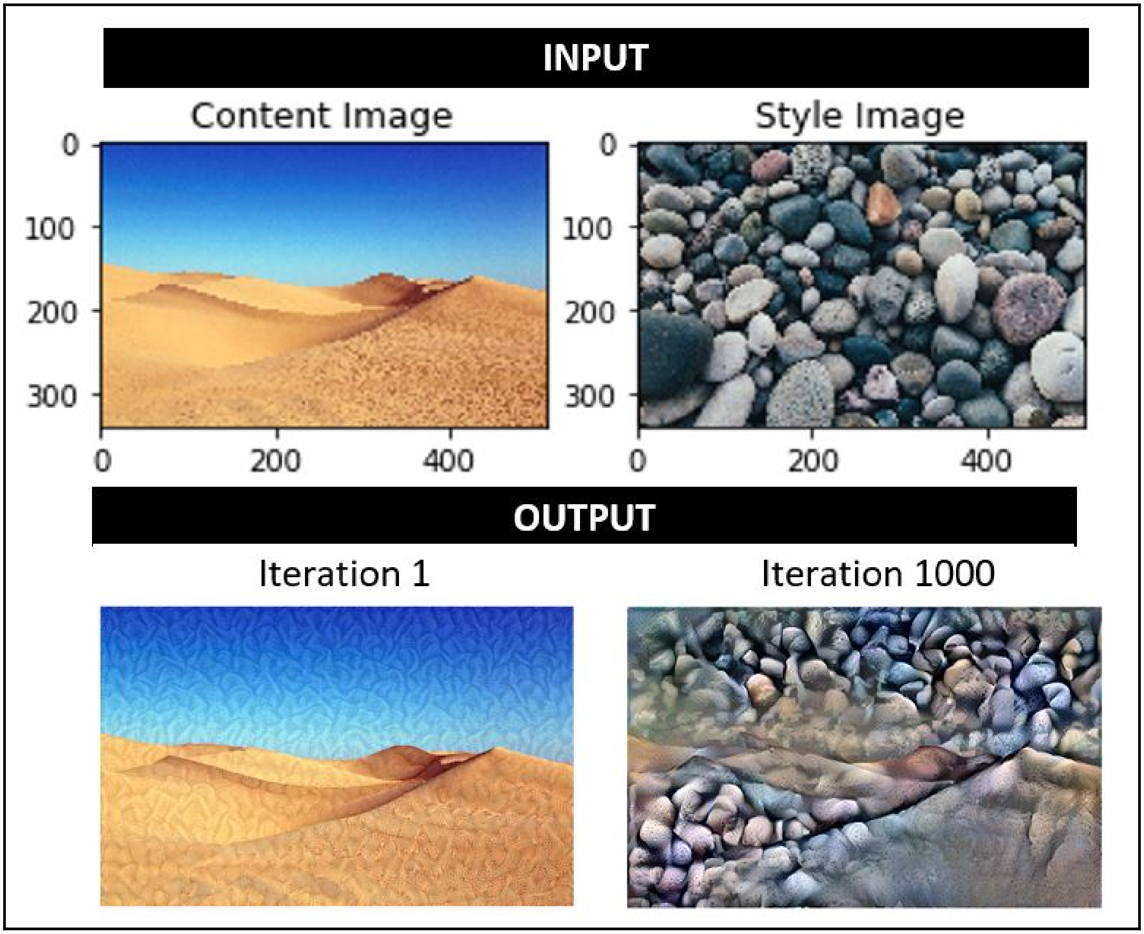

input image
segmentation map
segmentation overlay
background
bicycle
input image
segmentation map
segmentation overlay
background
bus
car
input image
segmentation map
segmentation overlay
background
bus
car
person
input image
segmentation map
segmentation overlay
background
chair
diningtable
sofa
tv


Dense layer (7,7,256)
Batch normalization
Leaky ReLU
Conv2D T, Stride = 1
Batch normalization
Leaky ReLU
Conv2D T, Stride = 2
Batch normalization
Leaky ReLU
Conv2D T, Stride = 2
∑
Generator Model

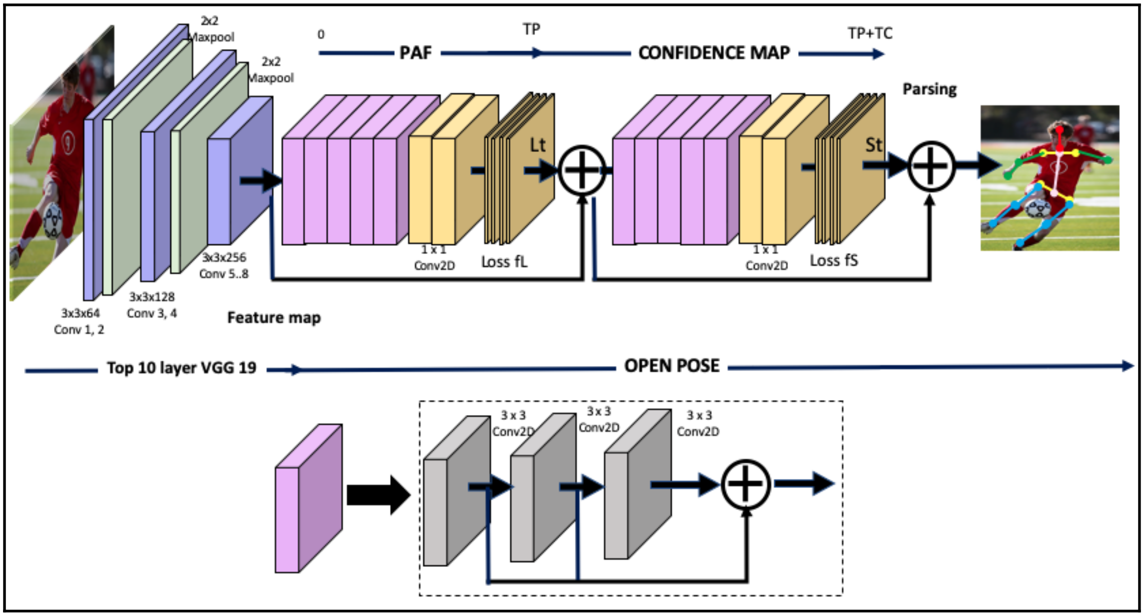

Conv2D, Stride = 1
Conv2D, Stride = 2
Batch normalization
Leaky ReLU
Dropout = 0.3
Conv2D, Stride = 2
Batch normalization
Leaky ReLU
Dropout = 0.3
∑
Flatten
Dense (1)
Discriminator Model

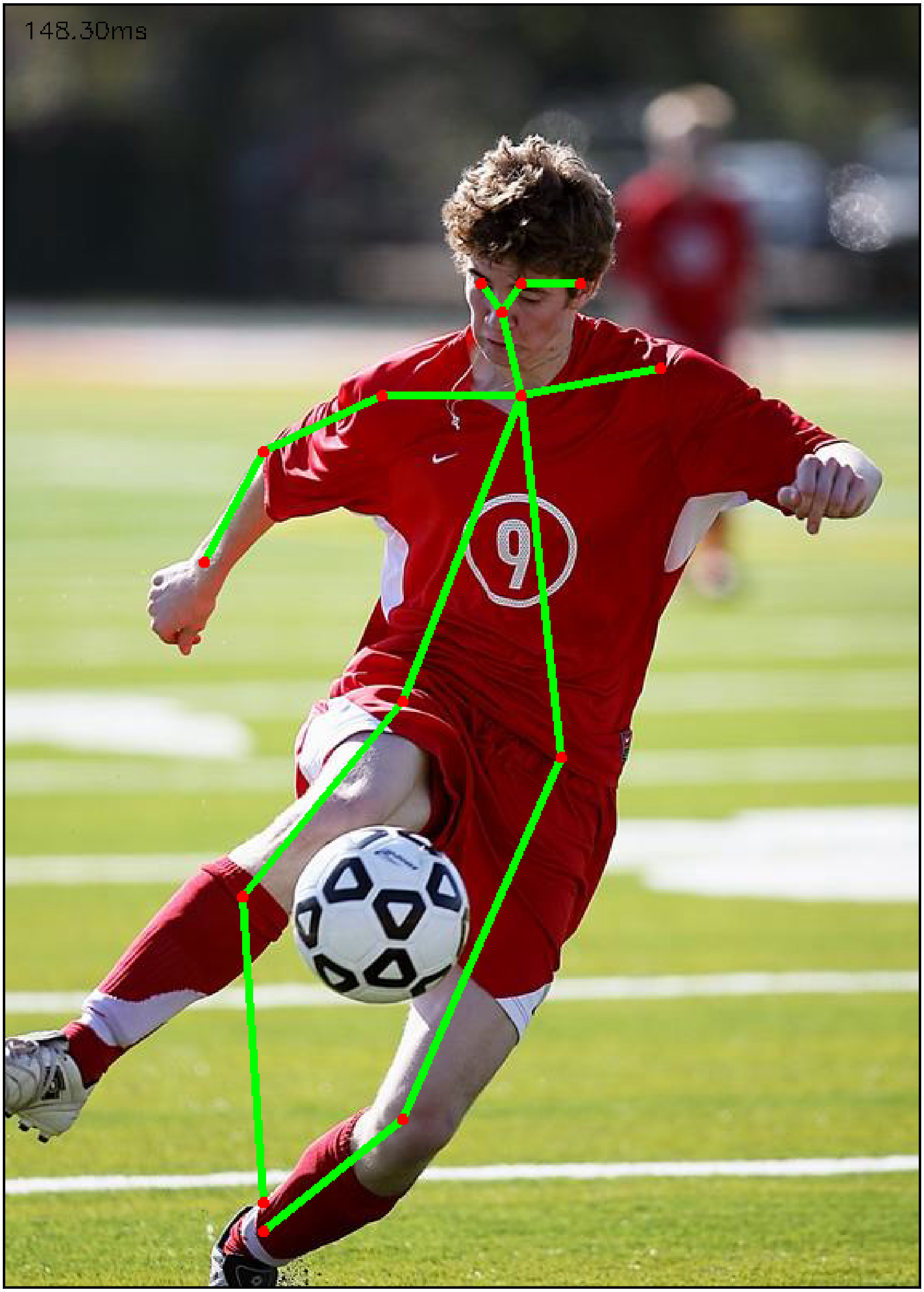

Cross – entropy (prediction of real image to array of 1)
∑
Cross – entropy (prediction of fake image to array of 0)
Discriminator Loss
Cross – entropy (prediction of fake image to array of 1)
Generator Loss


Input image
Mask image
Output - Method 1
Output - Method 2

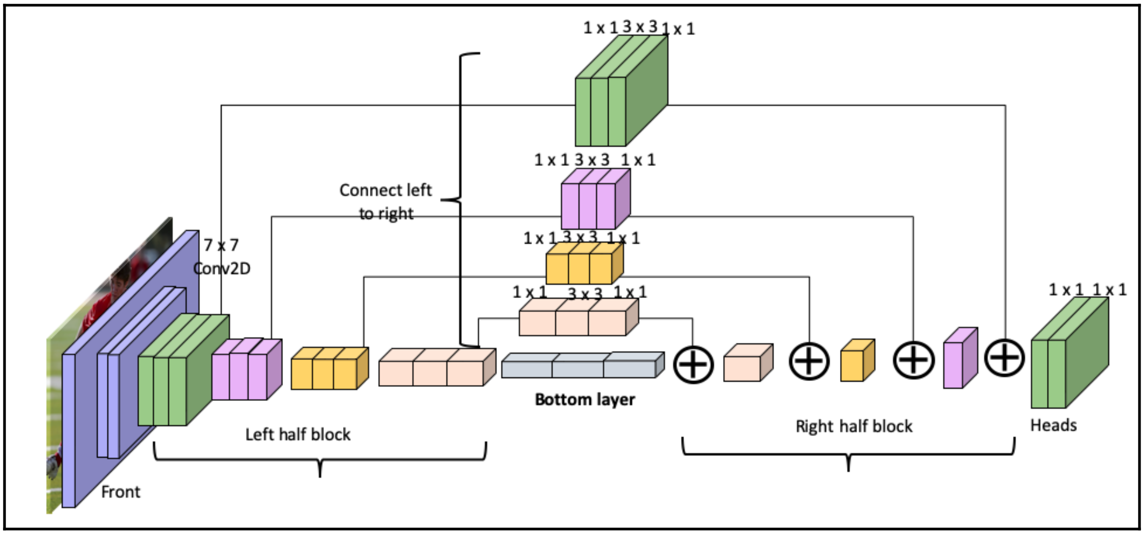

INPUT
Content Image
Style Image
0
100
200
300
0
200
400
OUTPUT
Iteration 1
Iteration 1000

Chapter 9: Action Recognition Using Multitask Deep Learning

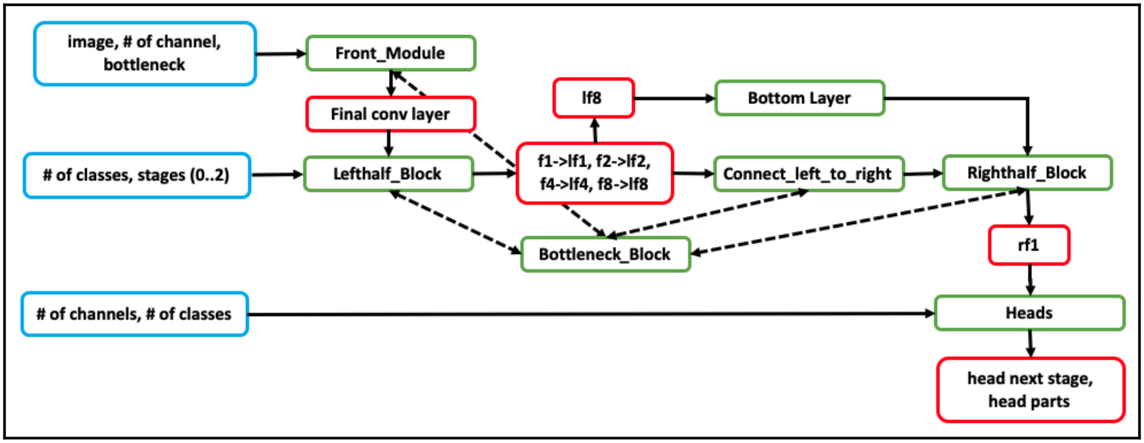


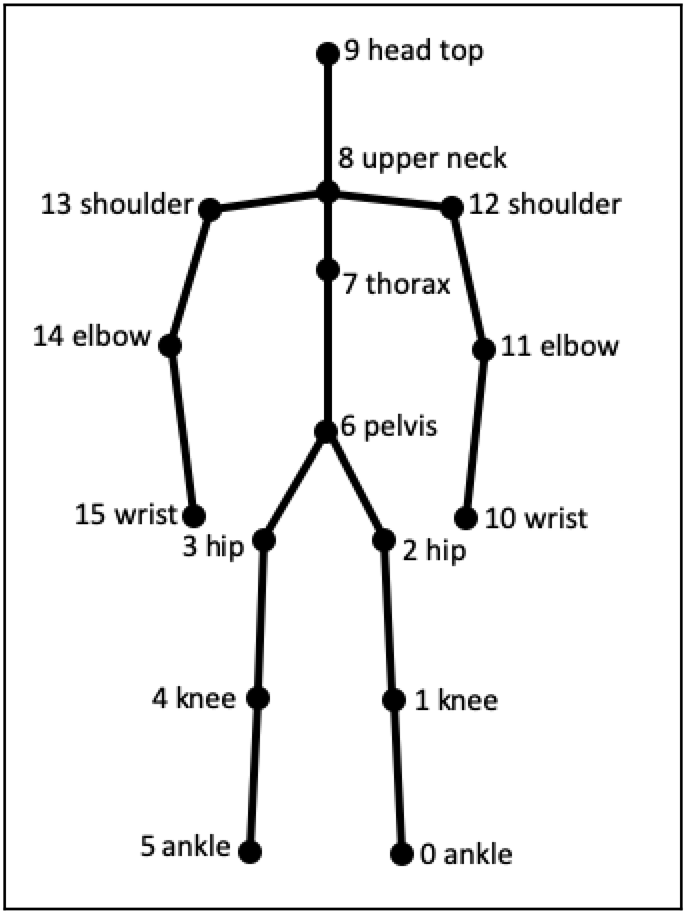

148.30ms

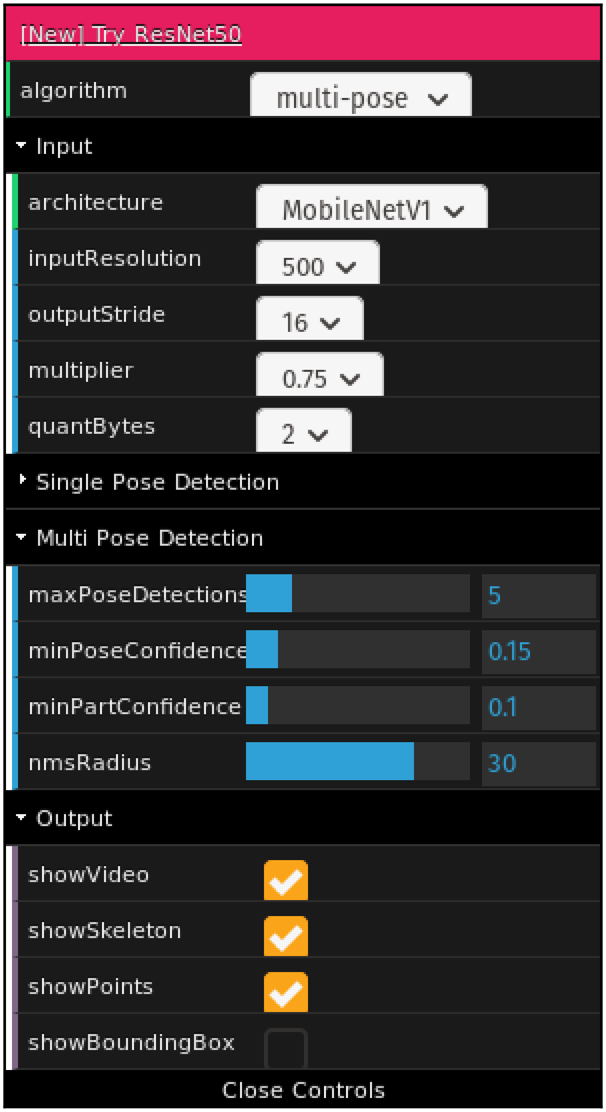
Baseball with background
Baseball without background

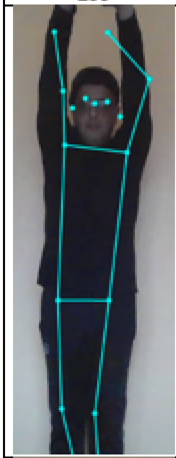
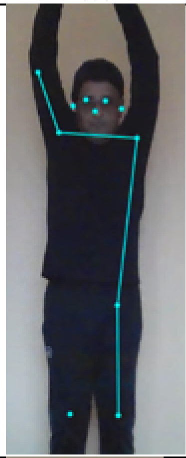
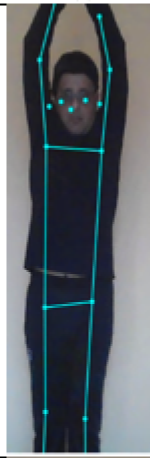
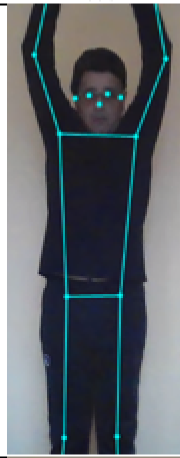
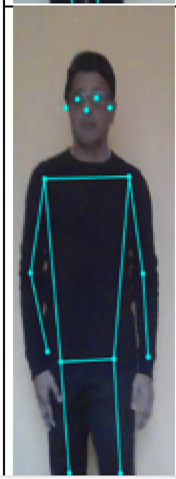
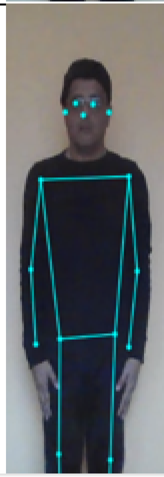
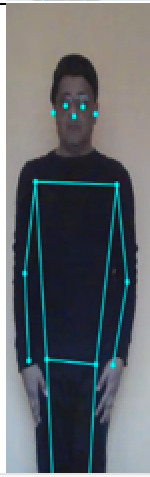
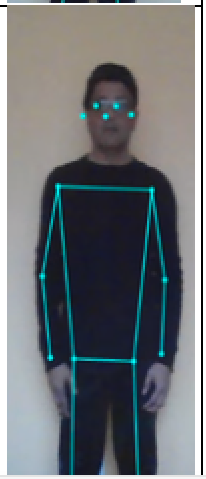


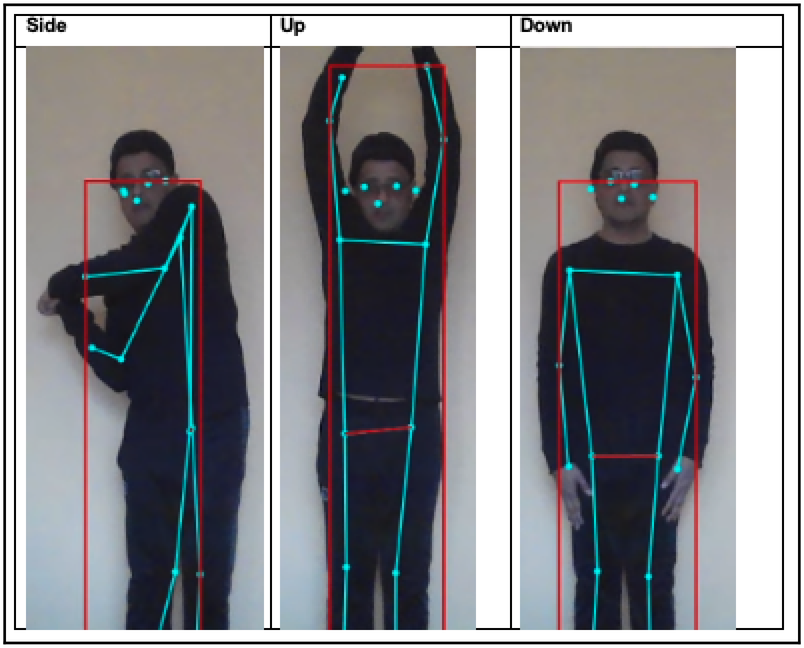
1 x 1 3 x 3 1 x 1
1 x 1 3 x 3 1 x 1
Connect left to right
7 x 7 Conv2D
1 x 1 3 x 3 1 x 1
1 x 1 3 x 3 1 x 1
1 x 1 1 x 1
Bottom layer
Left half block
Right half block
Heads
Front

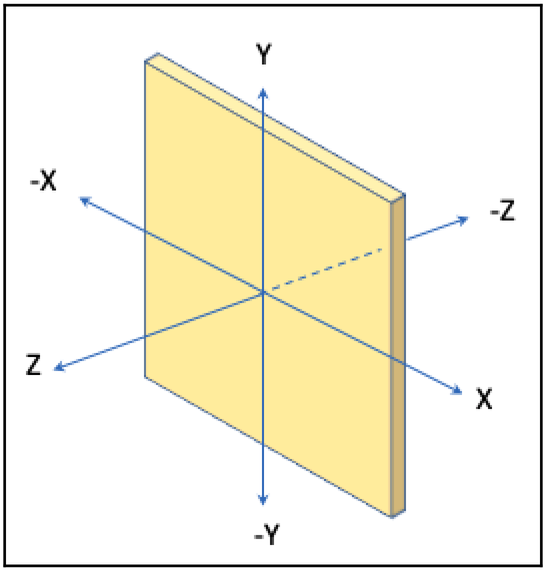

image, # of channel, bottleneck
Front_Module
Final conv layer
lf8
Bottom Layer
# of classes, stages (0..2)
Lefthalf_Block
f1->lf1, f2->lf2, f4->lf4, f8->lf8
Connect_left_to_right
Righthalf_Block
Bottleneck_Block
rf1
# of channels, # of classes
Heads
head next stage, head parts


9 head top
8 upper neck
13 shoulder
12 shoulder
7 thorax
14 elbow
11 elbow
6 pelvis
15 wrist
10 wrist
3 hip
2 hip
4 knee
1 knee
5 ankle
0 ankle


[New] Try ResNet50
algorithm
multi-pose
Input
architecture
MobileNetV1
inputResolution
500
outputStride
16
multiplier
0.75
quantBytes
2
Single Pose Detection
Multi Pose Detection
maxPoseDetections
5
minPoseConfidence
0.15
minPartConfidence
0.1
nmsRadius
30
Output
showVideo
showSkeleton
showPoints
showBoundingBox
Close Controls

MobileNet		ResNet	
200	500	200	500
			
			


Side
Up
Down


Y
-X
-Z
Z
X
-Y

Softmax, split at 18

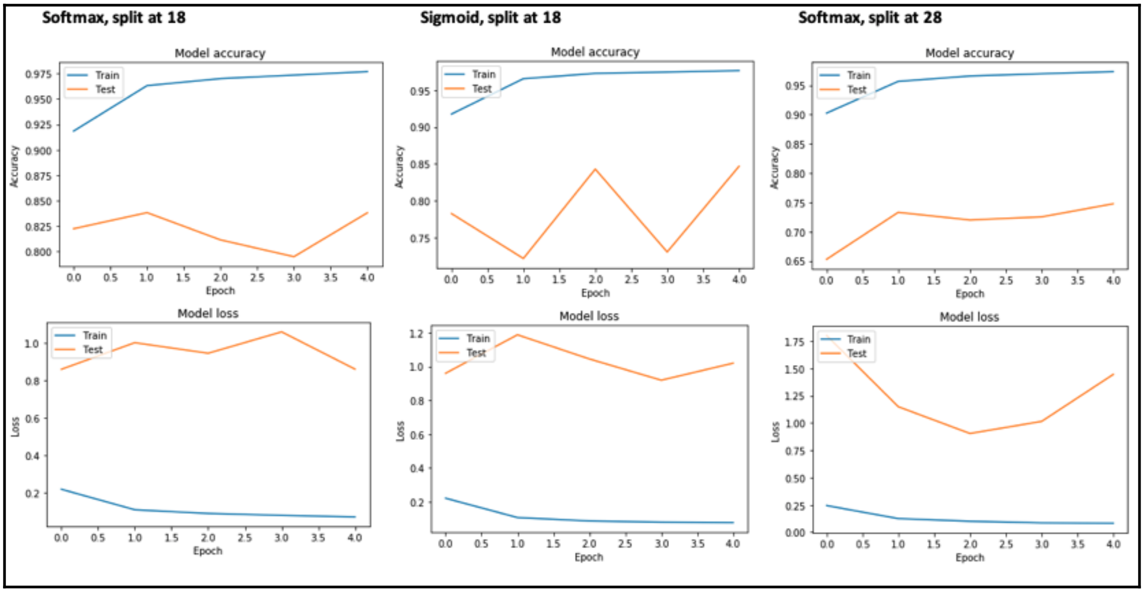
```
Train on 12018 steps, validate on 17987 steps
Epoch 1/5
12018/12018 [==============================] - 40s 3ms/step - loss: 0.2205 - acc: 0.9184 - val_loss: 0.8585 - val_acc: 0.8224
Epoch 2/5
12018/12018 [==============================] - 39s 3ms/step - loss: 0.1113 - acc: 0.9631 - val_loss: 1.0003 - val_acc: 0.8382
Epoch 3/5
12018/12018 [==============================] - 39s 3ms/step - loss: 0.0917 - acc: 0.9700 - val_loss: 0.9439 - val_acc: 0.8114
Epoch 4/5
12018/12018 [==============================] - 39s 3ms/step - loss: 0.0815 - acc: 0.9734 - val_loss: 1.0577 - val_acc: 0.7949
Epoch 5/5
12018/12018 [==============================] - 39s 3ms/step - loss: 0.0732 - acc: 0.9768 - val_loss: 0.8594 - val_acc: 0.8380
```

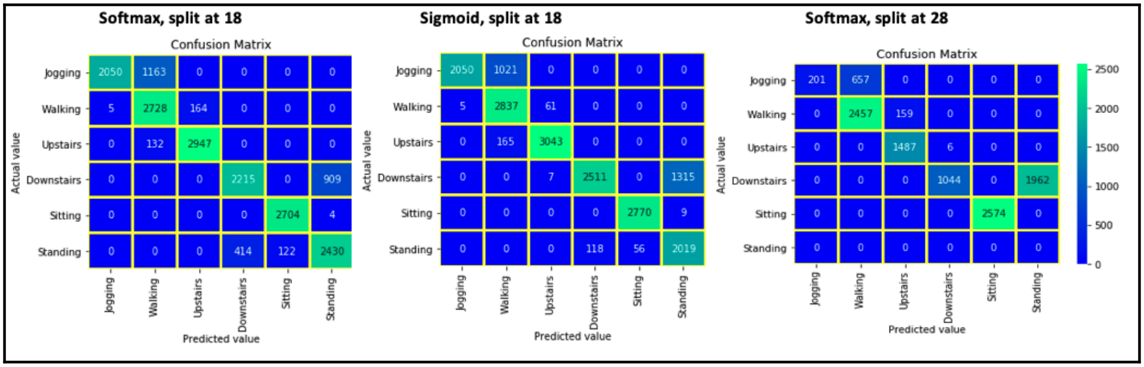
Sigmoid, split at 18

```
Train on 12018 steps, validate on 17987 steps
Epoch 1/5
12018/12018 [==============================] - 41s 3ms/step - loss: 0.2211 - acc: 0.9175 - val_loss: 0.9594 - val_acc: 0.7825
Epoch 2/5
12018/12018 [==============================] - 40s 3ms/step - loss: 0.1068 - acc: 0.9655 - val_loss: 1.1869 - val_acc: 0.7216
Epoch 3/5
12018/12018 [==============================] - 40s 3ms/step - loss: 0.0866 - acc: 0.9726 - val_loss: 1.0440 - val_acc: 0.8428
Epoch 4/5
12018/12018 [==============================] - 40s 3ms/step - loss: 0.0794 - acc: 0.9745 - val_loss: 0.9185 - val_acc: 0.7304
Epoch 5/5
12018/12018 [==============================] - 40s 3ms/step - loss: 0.0770 - acc: 0.9764 - val_loss: 1.0192 - val_acc: 0.8467
```

Softmax, split at 28

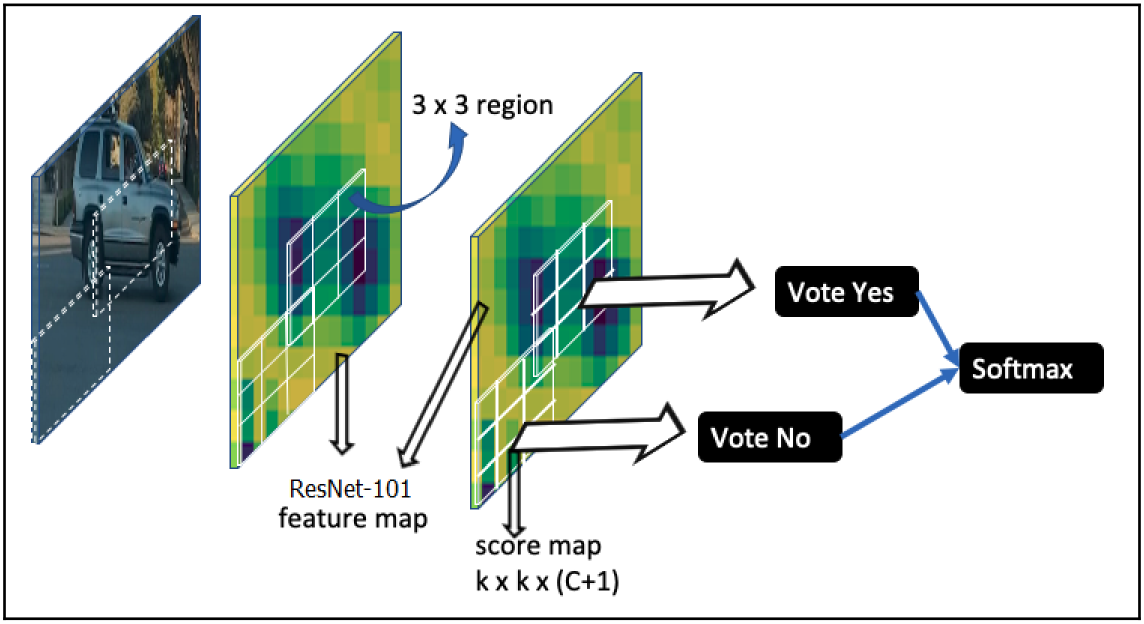
```
Train on 19458 steps, validate on 10547 steps
Epoch 1/5
19458/19458 [==============================] - 45s 2ms/step - loss: 0.2492 - acc: 0.9009 - val_loss: 1.5317 - val_acc: 0.6379
Epoch 2/5
19458/19458 [==============================] - 44s 2ms/step - loss: 0.1207 - acc: 0.9558 - val_loss: 1.2396 - val_acc: 0.6741
Epoch 3/5
19458/19458 [==============================] - 44s 2ms/step - loss: 0.1040 - acc: 0.9637 - val_loss: 1.2310 - val_acc: 0.6623
Epoch 4/5
19458/19458 [==============================] - 44s 2ms/step - loss: 0.0888 - acc: 0.9698 - val_loss: 2.8962 - val_acc: 0.6556
Epoch 5/5
19458/19458 [==============================] - 44s 2ms/step - loss: 0.0835 - acc: 0.9708 - val_loss: 1.7320 - val_acc: 0.7360
```

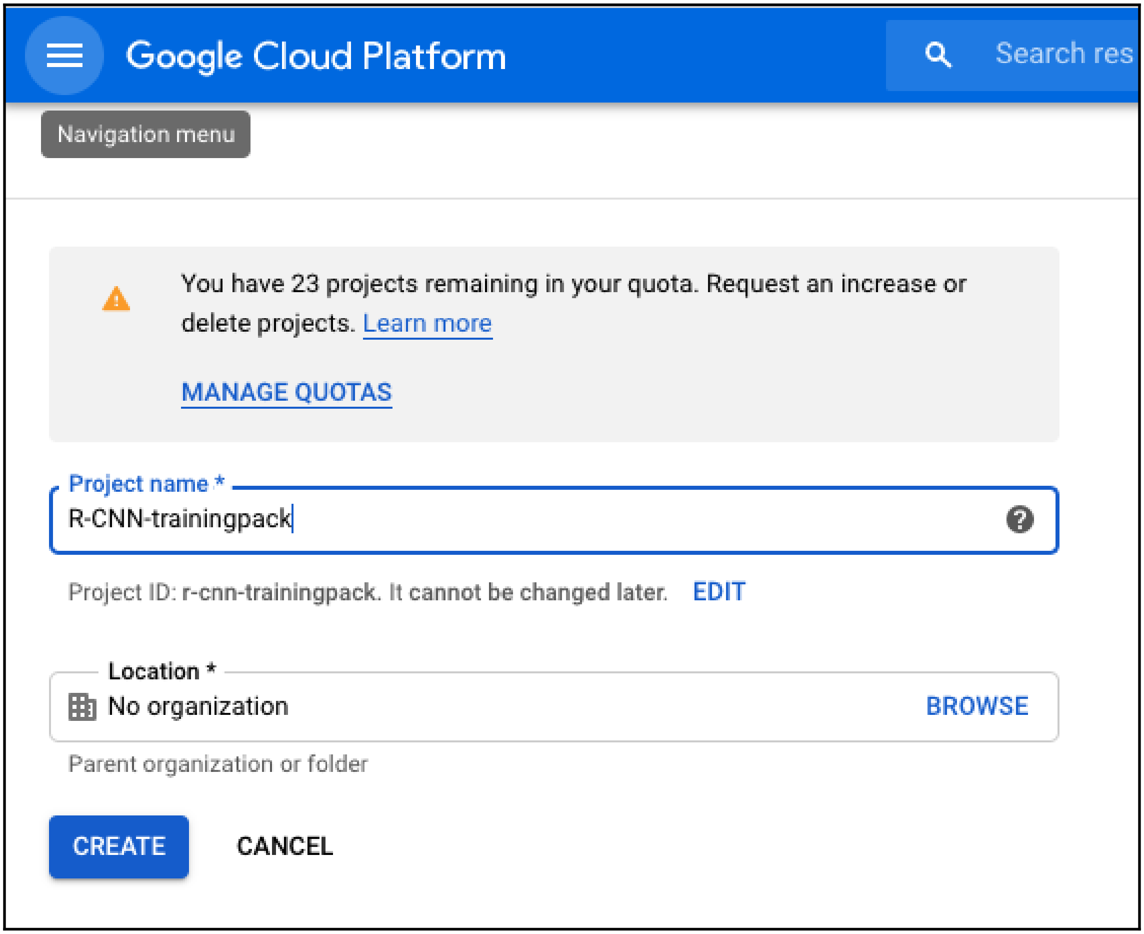


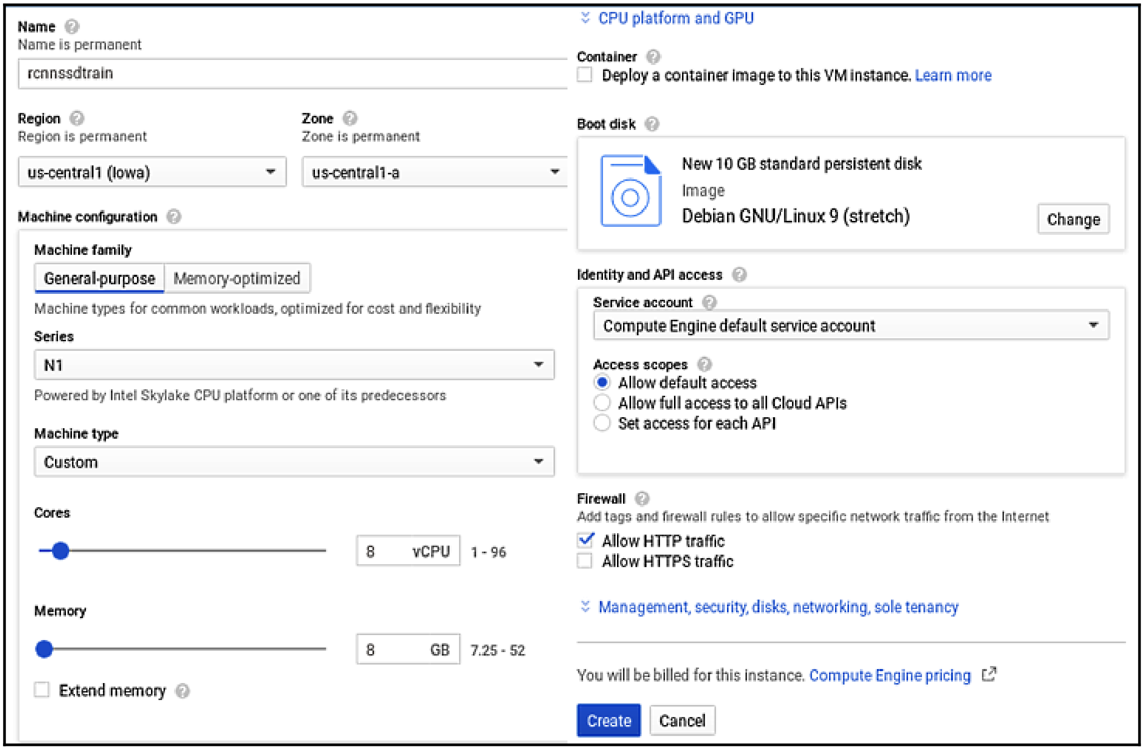


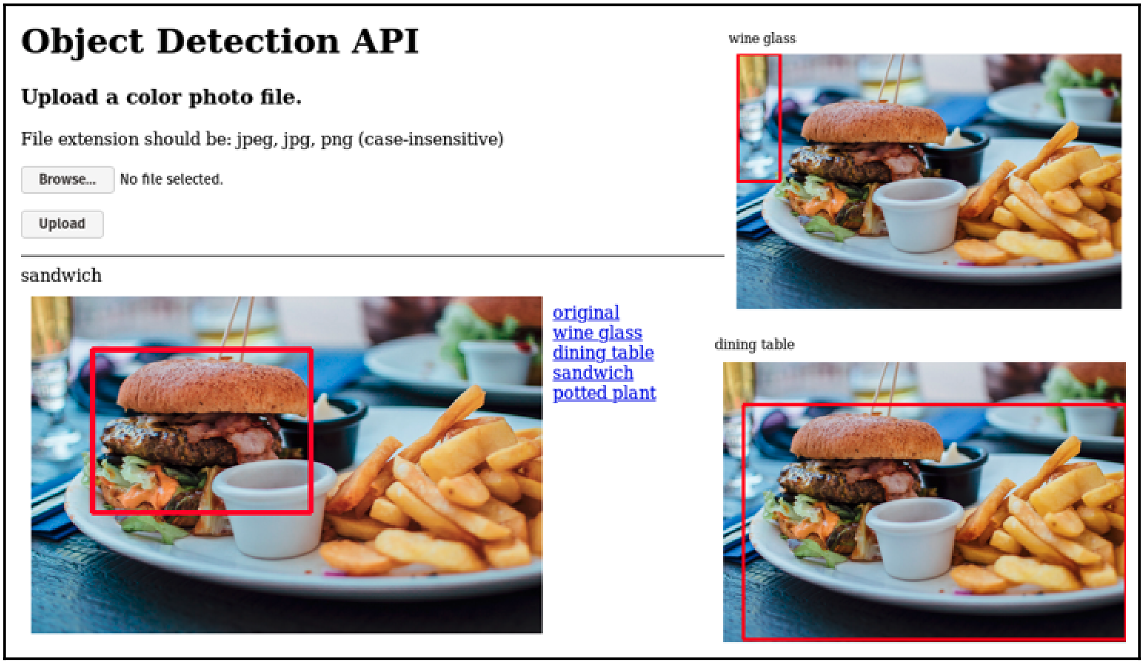
Softmax, split at 18
Sigmoid, split at 18
Softmax, split at 28
Confusion Matrix
Actual value
Predicted value
Jogging
Walking
Upstairs
Downstairs
Sitting
Standing

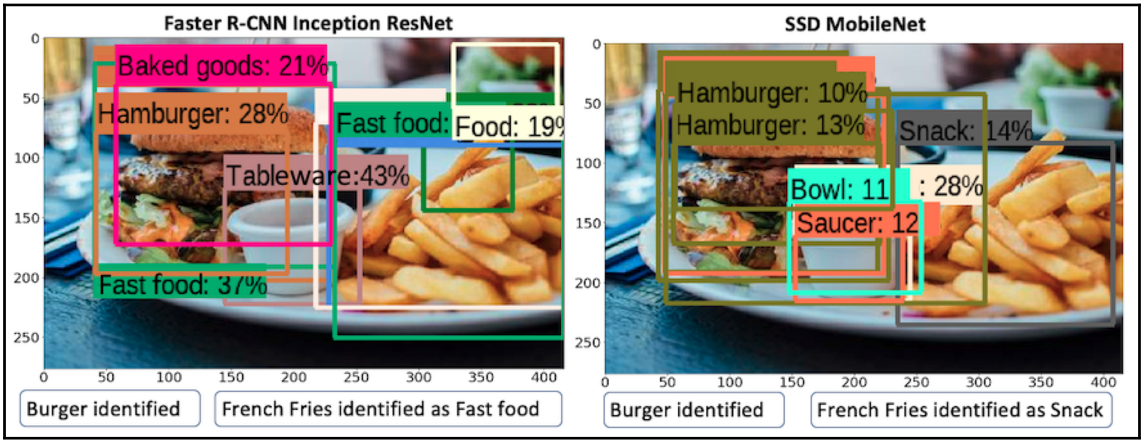
Chapter 10: Object Detection Using R-CNN, SSD, and R-FCN

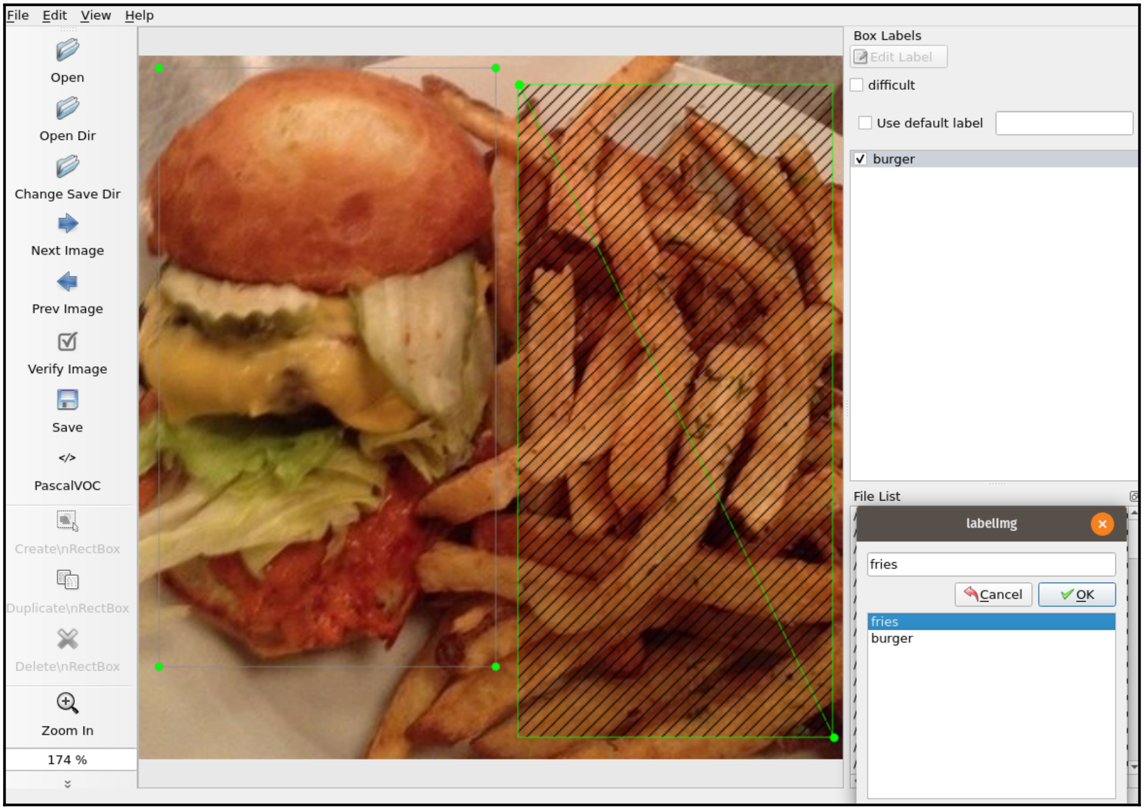



Google Cloud Platform
Search res
Navigation menu
You have 23 projects remaining in your quota. Request an increase or delete projects. Learn more
MANAGE QUOTAS
Project name *
R-CNN-trainingpack
Project ID: r-cnn-trainingpack. It cannot be changed later. EDIT
Location *
No organization
BROWSE
Parent organization or folder
CREATE
CANCEL

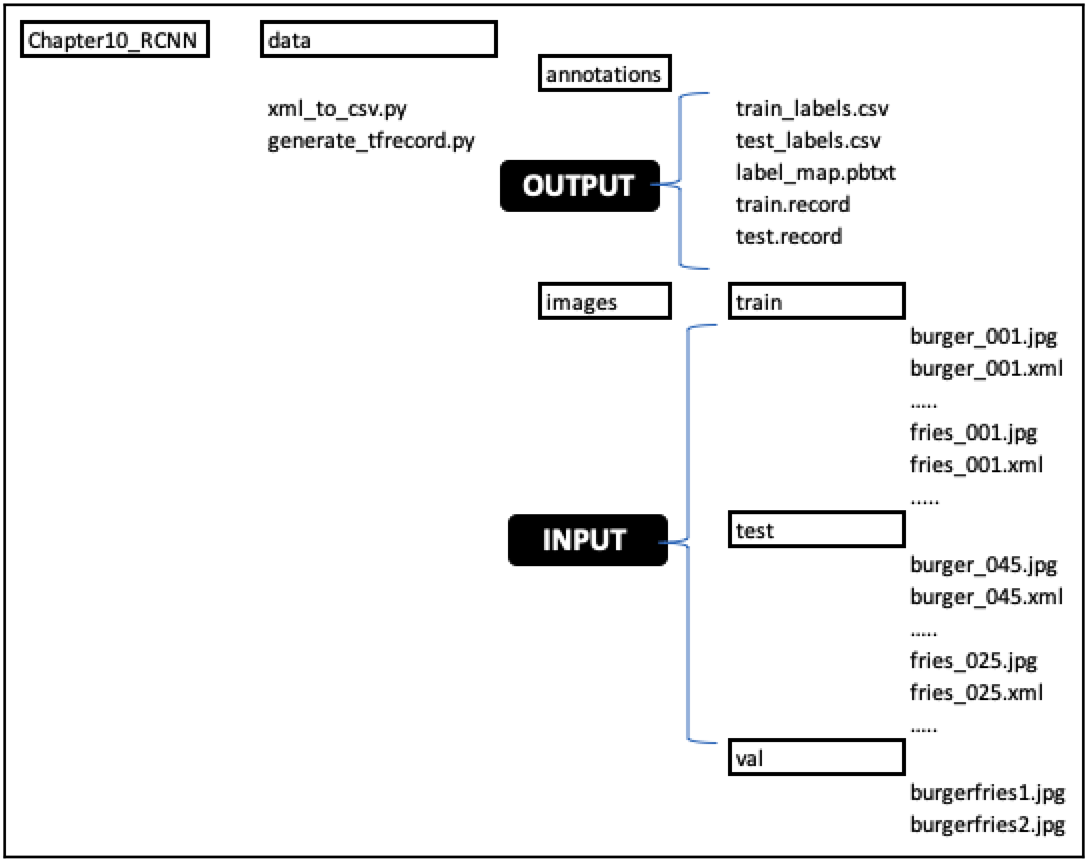

Name
Name is permanent
rcnnssdtrain
Region
Region is permanent
us-central1 (Iowa)
Zone
Zone is permanent
us-central1-a
Machine configuration
Machine family
General-purpose
Memory-optimized
Machine types for common workloads, optimized for cost and flexibility
Series
N1
Powered by Intel Skylake CPU platform or one of its predecessors
Machine type
Custom
Cores
8
vCPU
1 - 96
Memory
8
GB
7.25 - 52
Extend memory
CPU platform and GPU
Container
Deploy a container image to this VM instance. Learn more
Boot disk
New 10 GB standard persistent disk
Image
Debian GNU/Linux 9 (stretch)
Change
Identity and API access
Service account
Compute Engine default service account
Access scopes
Allow default access
Allow full access to all Cloud APIs
Set access for each API
Firewall
Add tags and firewall rules to allow specific network traffic from the Internet
Allow HTTP traffic
Allow HTTPS traffic
Management, security, disks, networking, sole tenancy
You will be billed for this instance. Compute Engine pricing
Create
Cancel

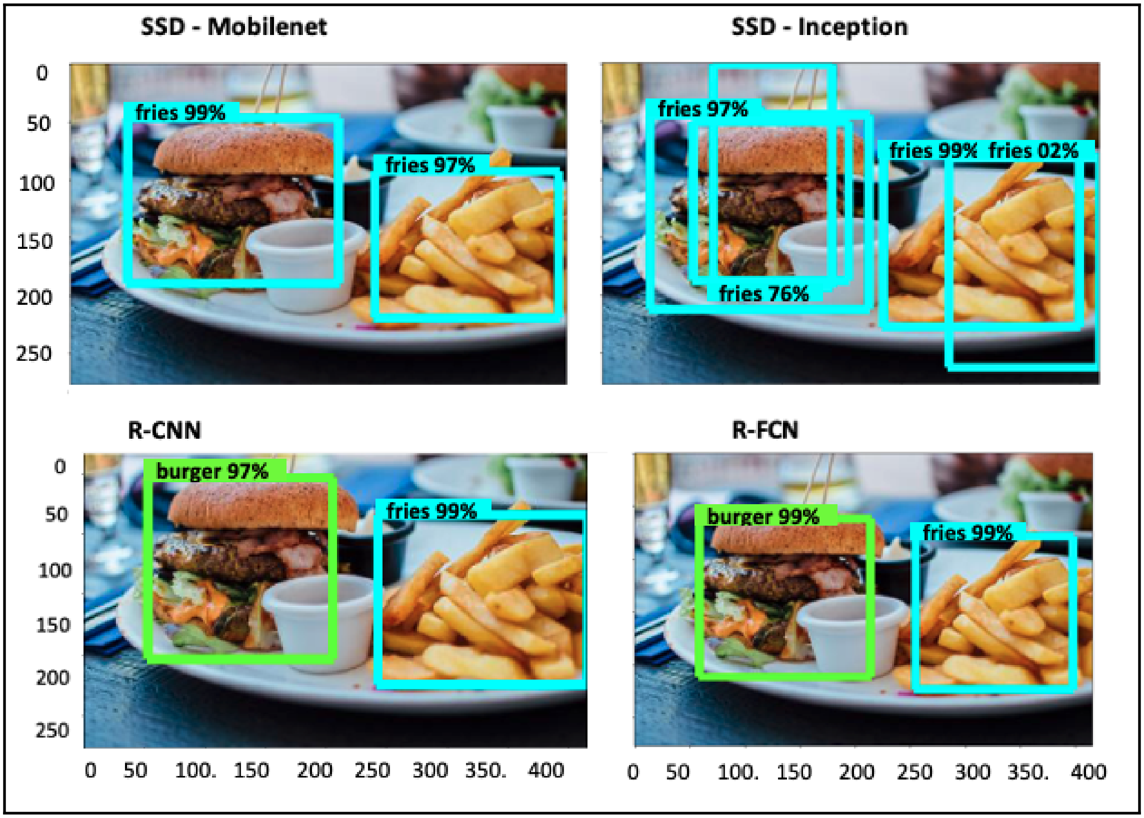

Object Detection API
Upload a color photo file.
File extension should be: jpeg, jpg, png (case-insensitive)
Browse... No file selected.
Upload
sandwich
original
wine glass
dining table
sandwich
potted plant
wine glass
dining table

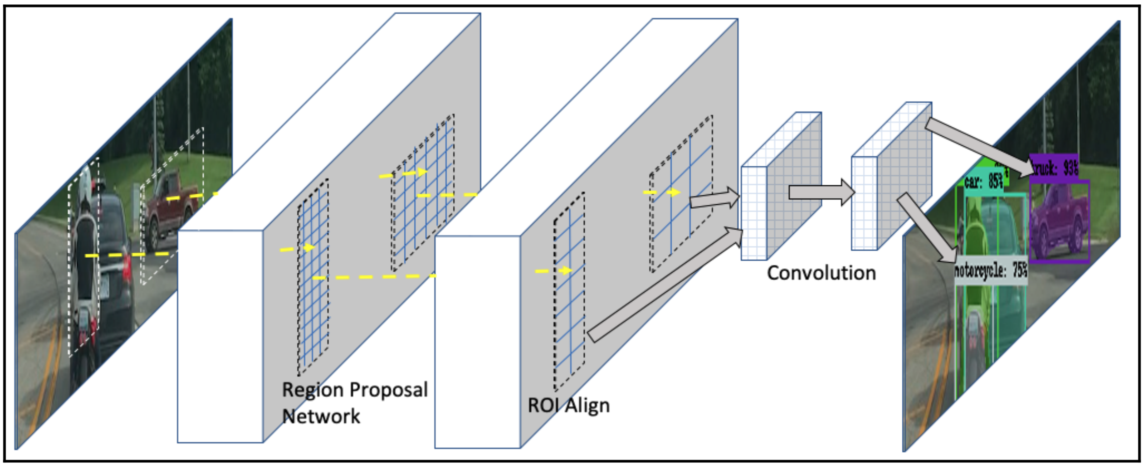

Faster R-CNN Inception ResNet
Baked goods: 21%
Hamburger: 28%
Fast food:
Tableware:43%
Fast food: 37%
Burger identified
French Fries identified as Fast food
SSD MobileNet
Hamburger: 10%
Hamburger: 13%
Snack: 14%
Bowl: 11
: 28%
Saucer: 12
Burger identified
French Fries identified as Snack

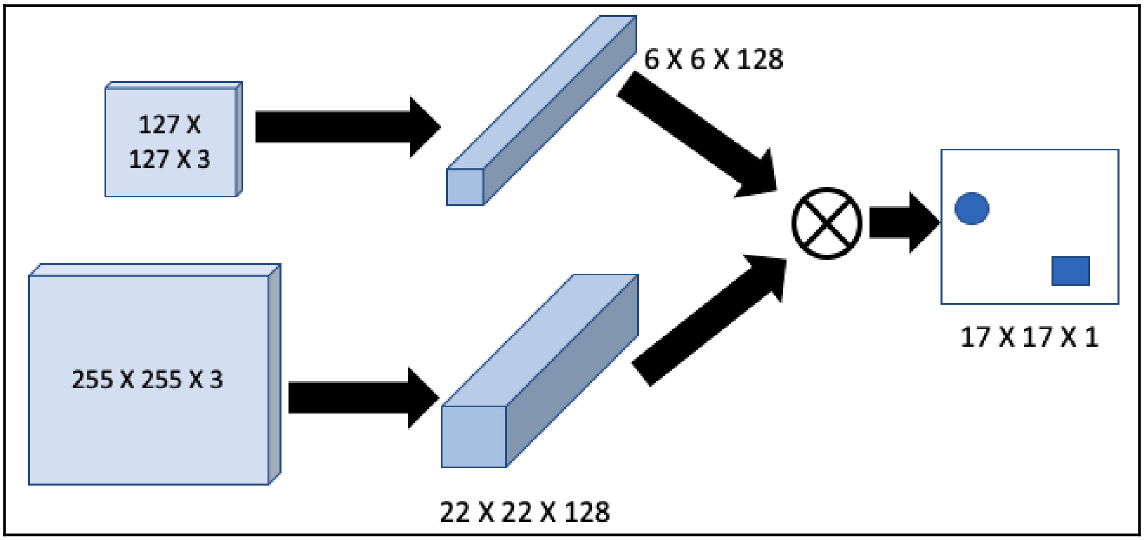

File Edit View Help
Open
Open Dir
Change Save Dir
Next Image
Prev Image
Verify Image
Save
</>
PascalVOC
Create\nRectBox
Duplicate\nRectBox
Delete\nRectBox
Zoom In
174 %
Box Labels
Edit Label
difficult
Use default label
burger
File List
labelImg
fries
Cancel
OK
fries
burger


My Drive > Chapter10_RCNN > data
Folders
annotations
images

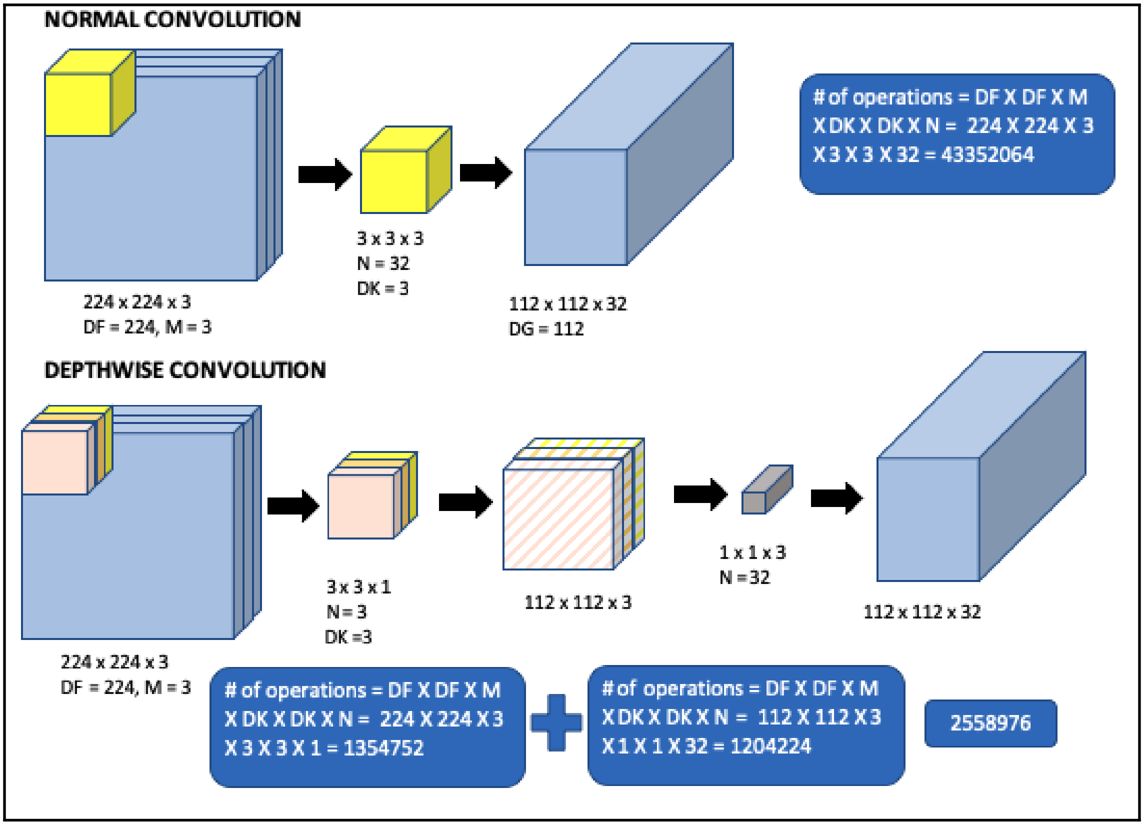

Chapter10_RCNN
data
annotations
xml_to_csv.py
generate_tfrecord.py
OUTPUT
train_labels.csv
test_labels.csv
label_map.pbtxt
train.record
test.record
images
train
burger_001.jpg
burger_001.xml
.....
fries_001.jpg
fries_001.xml
.....
INPUT
test
burger_045.jpg
burger_045.xml
.....
fries_025.jpg
fries_025.xml
.....
val
burgerfries1.jpg
burgerfries2.jpg


SSD - Mobilenet
SSD - Inception
fries 99%
fries 97%
fries 97%
fries 76%
fries 99%
fries 02%
R-CNN
R-FCN
burger 97%
fries 99%
burger 99%
fries 99%
0
50
100
150
200
250
0 50 100. 150 200 250 300 350. 400


Region Proposal Network
ROI Align
Convolution


6 X 6 X 128
127 X
127 X 3
255 X 255 X 3
22 X 22 X 128
17 X 17 X 1

Chapter 11: Deep Learning on Edge Devices with CPU/GPU Optimization




intel
Neural Compute Stick 2
HDMI

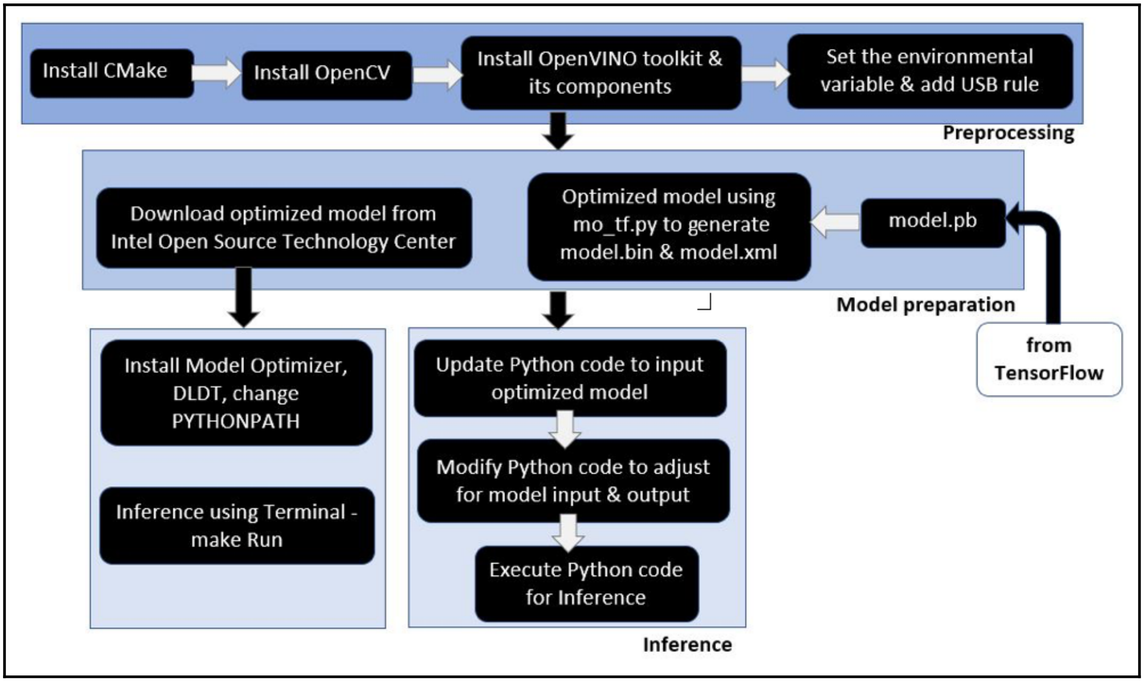

01 Intel® Open Source Technology Center
01.org Crosswalk Project Jira Privacy Policy Terms of Service
Name Last modified Size Descr
Parent Directory -
R1/ 2020-03-18 07:39 -
R2/ 2019-07-16 09:00 -
R3/ 2019-09-05 0
R4/ 2020-01-17 0
FP16/
FP32/
INT8/
Parent Directory
20190404_140900_models_bin/ 2019-04-05 07:15 -
models_bin/ 2019-05-06 10:30 -
models_bin_20190329_105530/ 2019-04-08 08:25 -
action-recognition-0001-decoder/
action-recognition-0001-encoder/
age-gender-recognition-retail-0013/
driver-action-recognition-adas-0002-decoder/
driver-action-recognition-adas-0002-encoder/
emotions-recognition-retail-0003/
face-detection-adas-0001/
face-detection-adas-binary-0001/
face-detection-retail-0004/
face-person-detection-retail-0002/
face-reidentification-retail-0095/
facial-landmarks-35-adas-0002/
gaze-estimation-adas-0002/
head-pose-estimation-adas-0001/
human-pose-estimation-0001/
instance-segmentation-security-0033/
instance-segmentation-security-0049/
landmarks-regression-retail-0009/
license-plate-recognition-barrier-0001/
person-attributes-recognition-crossroad-0230/
person-detection-action-recognition-0005/
person-detection-action-recognition-teacher-0002/
person-detection-retail-0002/
person-detection-retail-0013/
person-reidentification-retail-0031/
person-reidentification-retail-0076/
person-reidentification-retail-0079/
person-vehicle-bike-detection-crossroad-0078/
resnet50-binary-0001/
road-segmentation-adas-0001/
semantic-segmentation-adas-0001/
single-image-super-resolution-1032/
single-image-super-resolution-1033/
text-detection-0002/
text-recognition-0012/
vehicle-attributes-recognition-barrier-0039/
vehicle-detection-adas-0002/
vehicle-detection-adas-binary-0001/
vehicle-license-plate-detection-barrier-0106/

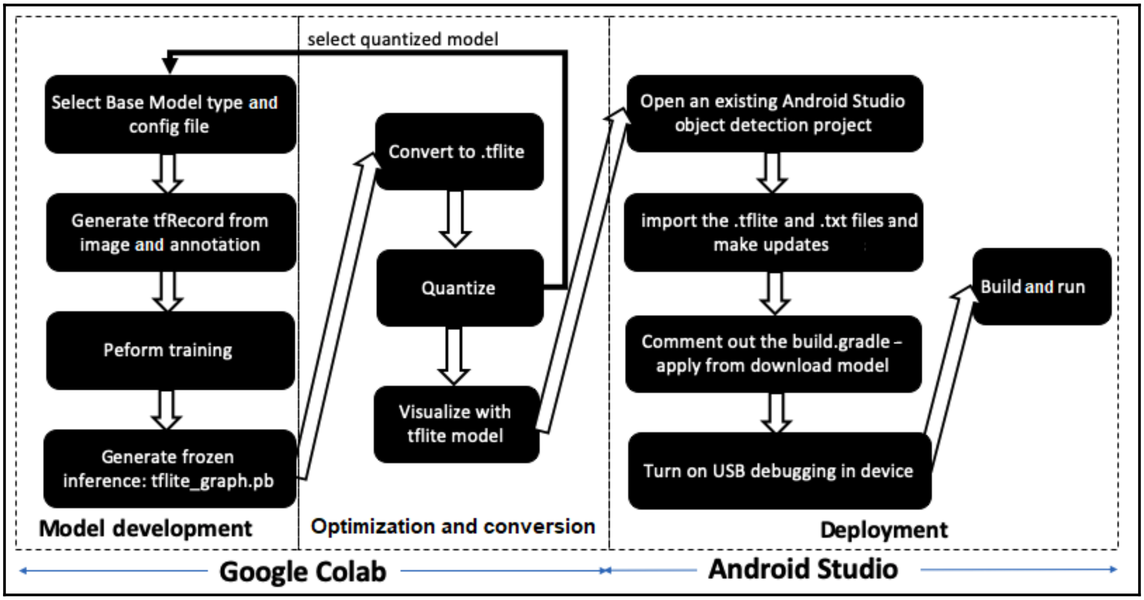
Model	Category	Input	Output	Note on output class
face-detection-adas-0001	face detection	image: [1xCxHxW]- shape [1x3x384x672]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
face-detection-adas-binary-0001	face detection	image: [1xCxHxW]- shape [1x3x384x672]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
face-detection-retail-0004	face detection	image: [1xCxHxW]- shape [1x3x300x300]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
face-detection-retail-0005	face detection	image: [1xCxHxW]- shape [1x3x300x300]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
person-detection-retail-0013	person & car detection	image: [1xCxHxW]- shape [1x3x320x544]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
pedestrian-detection-adas-0002	person & car detection	image: [1xCxHxW]- shape [1x3x384x672]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
pedestrian-detection-adas-binary-0001	person & car detection	image: [1xCxHxW]- shape [1x3x384x672]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
pedestrian-and-vehicle-detector-adas-0001	person & car detection	image: [1xCxHxW]- shape [1x3x384x672]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
vehicle-detection-adas-0002	person & car detection	image: [1xCxHxW]- shape [1x3x384x672]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
vehicle-detection-adas-binary-0001	person & car detection	image: [1xCxHxW]- shape [1x3x384x672]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
person-vehicle-bike-detection-crossroad-0078	person & car detection	image: [1xCxHxW]- shape [1x3x1024x1024]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
person-vehicle-bike-detection-crossroad-1016	person & car detection	image: [1xCxHxW]- shape [1x3x512x512]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
vehicle-license-plate-detection-barrier-0106	license plate	image: [1xCxHxW]- shape [1x3x300x300]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]

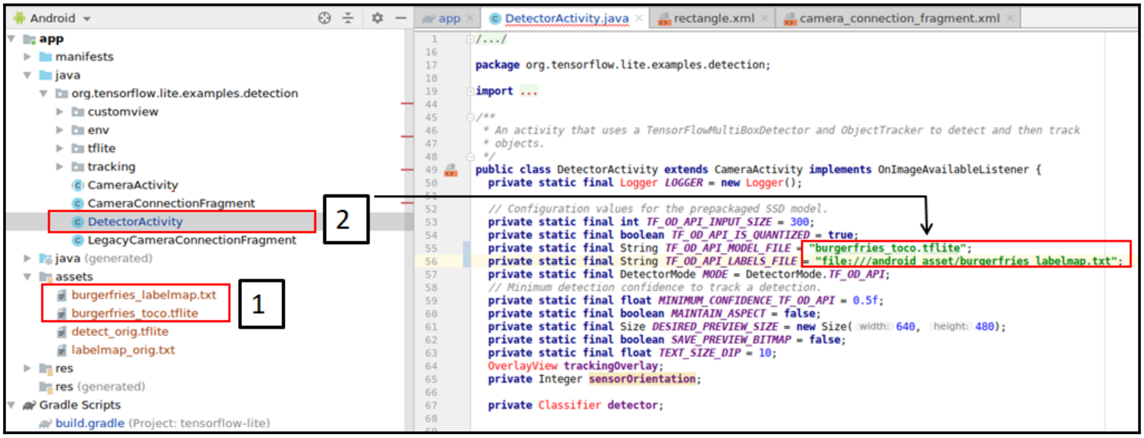
Model	Category	Input	Output	Note on output class
age-gender-recognition-retail-0013	attribute	image: [1xCxHxW]- shape [1x3x60x60]	"age_conv3", prob	age = age_conv3 *100
vehicle-attributes-recognition-barrier-0039	attribute	image: [1xCxHxW]- shape [1x3x72x72]	color, type	color classes [white, gray, yellow, red, green, blue, black], type classes [car, bus, truck, van]
emotion-recognition-retail-0003	attribute	image: [1xCxHxW]- shape [1x3x64x64]	prob	five emotions ('neutral', 'happy', 'sad', 'surprise', 'anger').
landmarks-regression-retail-0009	landmark	image: [1xCxHxW]- shape [1x3x48x48]	blob of shape [1,10]	5 landmarks normalized coordinates in the form (x0, y0, x1, y1, ..., x4, y4). Actual x value = normalized value* bounding box width, Actual y value = normalized value* bounding box height
facial-landmarks-35-adas-0002	landmark	image: [1xCxHxW]- shape [1x3x60x60]	blob of shape [1,70]	35 landmarks normalized coordinates in the form (x0, y0, x1, y1, ..., x34, y34). Actual x value = normalized value* bounding box width, Actual y value = normalized value* bounding box height
person-attributes-recognition-crossroad-0230	attribute	image: [1xCxHxW]- shape [1x3x160x80]	blob 453, blob 456, blob 459	blob 453 has 8 attributes: [is_male, has_bag, has_backpack, has_hat, has_longsleeves, has_longpants, has_longhair, has_coat_jacket], blob 456 - top color, blob 459 bottom color

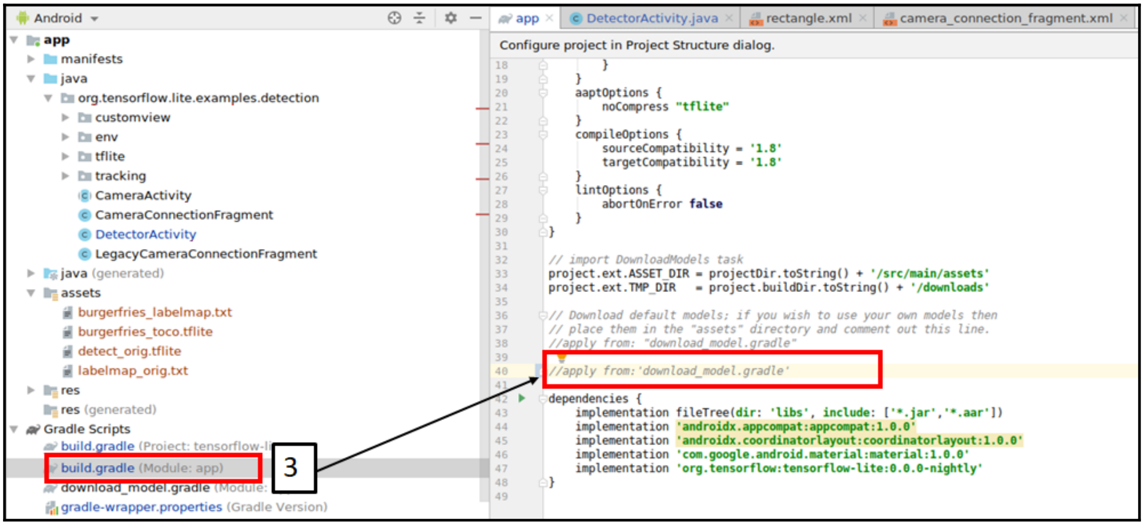
Model	Category	Input	Output	Note on output class
head-pose-estimation-adas-0001	pose estimation	image: [1xCxHxW]- shape [1x3x60x60]	angle_y_fc, angle_p_fc, angle_r_fc	yw, pitch, roll
person-detection-action-recognition-0005	action recognition	image: [1xCxHxW]- shape [1x3x400x680]	box coordinates in SSD format, detection confidence, prior box in SSD format, action confidence (anchor 1, 2, 3, 4)	
person-detection-action-recognition-0006	action recognition	image: [1xCxHxW]- shape [1x400x680x3]	box coordinates in SSD format, detection confidence, action confidence (anchor 1, 2, 3, 4, 5)	
person-detection-action-recognition-teacher-0002	action recognition	image: [1xCxHxW]- shape [1x3x400x680]	box coordinates in SSD format, detection confidence, prior box in SSD format, action confidence (anchor 1, 2, 3, 4)	
person-detection-raisinghand-recognition-0001	action recognition	image: [1xCxHxW]- shape [1x3x400x680]	box coordinates in SSD format, detection confidence, prior box in SSD format, action confidence (anchor 1, 2, 3, 4)	

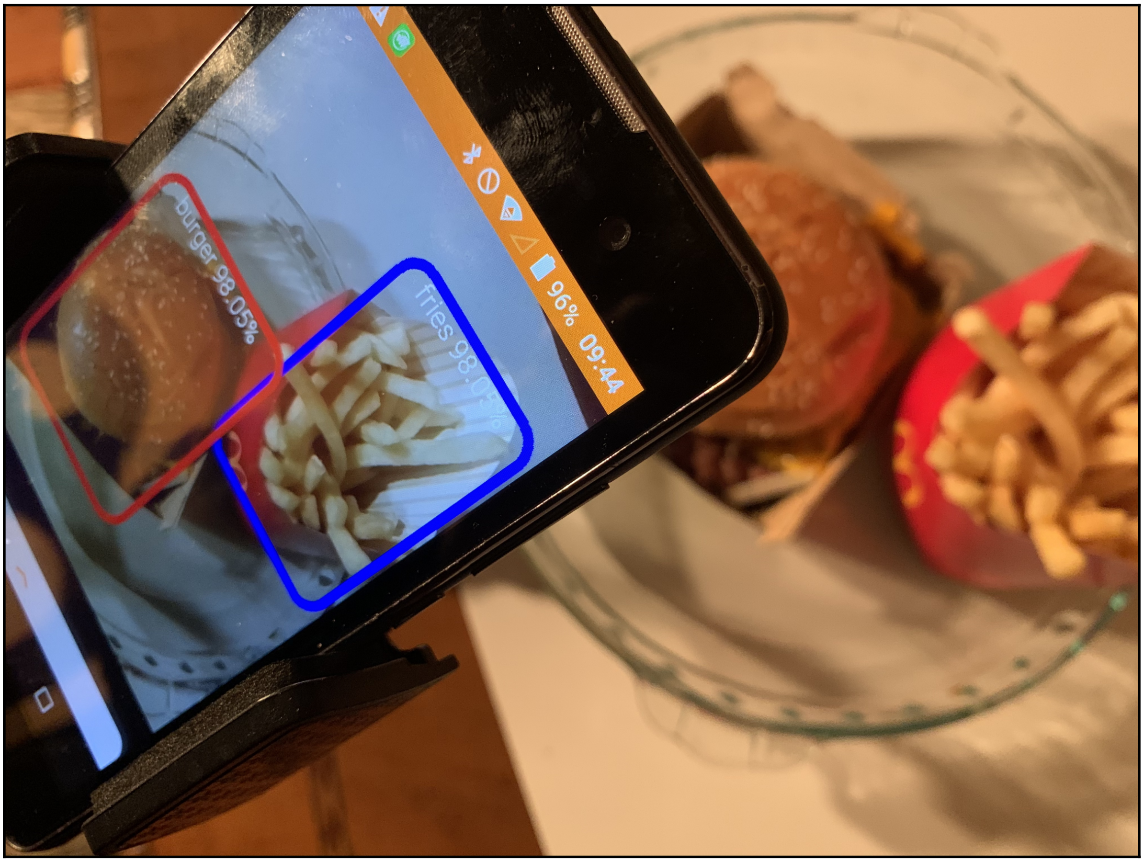
Model	Category	Input	Output	Note on output class
license-plate-recognition-barrier-0001	license plate - 2 input	image: [1xCxHxW]- shape [1x3x24x94]; seq_ind - set this to [0, ,1, ,1 .. 1] 88 values	"decode", shape: [1, 88, 1, 1]	
person-detection-retail-0002	person detection 2 input	Two inputs: 1) image: [1xCxHxW]-shape [1x3x544x992] 2) image_info shape [1,6] [544, 992, 992/frame_width, 544/frame_height, 992/frame_width, 544/frame_height]	blob [1, 1, N, 7]	[image_id, label, conf, x_min, y_min, x_max, y_max]
gaze-estimation-adas-0002	gaze estimation 3 input	three input blobs: 1) left eye image - [1xCxHxW] shape [1,3,60,60], 2) right eye image - [1xCxHxW] shape [1,3,60,60] 3) head pose angle - [BXC] shape [1,3]	gaze direction vector	output not normalized

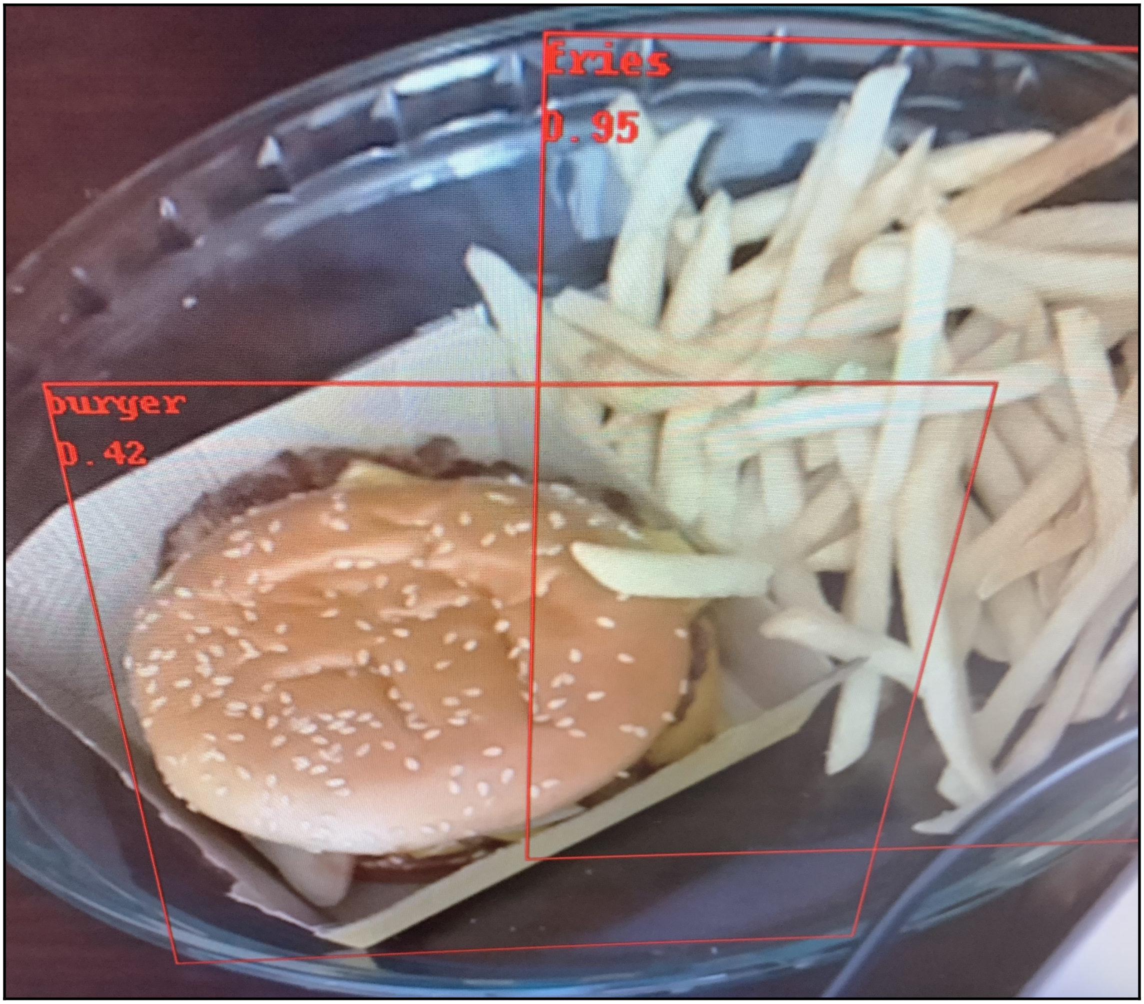

Install CMake
Install OpenCV
Install OpenVINO toolkit & its components
Set the environmental variable & add USB rule
Preprocessing
Download optimized model from Intel Open Source Technology Center
Optimized model using mo_tf.py to generate model.bin & model.xml
model.pb
Model preparation
from TensorFlow
Install Model Optimizer, DLDT, change PYTHONPATH
Inference using Terminal - make Run
Update Python code to input optimized model
Modify Python code to adjust for model input & output
Execute Python code for Inference
Inference


select quantized model
Select Base Model type and config file
Generate tfRecord from image and annotation
Peform training
Generate frozen inference: tflite_graph.pb
Model development
Convert to .tflite
Quantize
Visualize with tflite model
Optimization and conversion
Open an existing Android Studio object detection project
import the .tflite and .txt files and make updates
Comment out the build.gradle – apply from download model
Turn on USB debugging in device
Build and run
Deployment
Google Colab
Android Studio

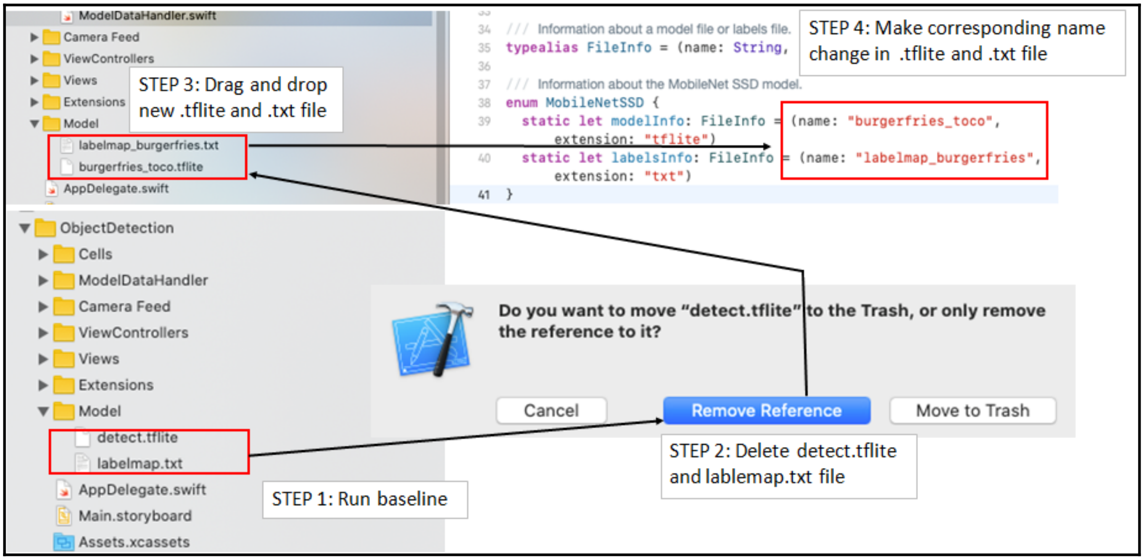

Android
app
manifests
java
org.tensorflow.lite.examples.detection
customview
env
tflite
tracking
CameraActivity
CameraConnectionFragment
DetectorActivity
LegacyCameraConnectionFragment
java (generated)
assets
burgerfries_labelmap.txt
burgerfries_toco.tflite
detect_orig.tflite
labelmap_orig.txt
res
res (generated)
Gradle Scripts
build.gradle (Project: tensorflow-lite)
1
2
app
DetectorActivity.java
rectangle.xml
camera_connection_fragment.xml
package org.tensorflow.lite.examples.detection;
import ...
* An activity that uses a TensorFlowMultiBoxDetector and ObjectTracker to detect and then track
* objects.
public class DetectorActivity extends CameraActivity implements OnImageAvailableListener {
private static final Logger LOGGER = new Logger();
// Configuration values for the prepackaged SSD model.
private static final int TF_OD_API_INPUT_SIZE = 300;
private static final boolean TF_OD_API_IS_QUANTIZED = true;
private static final String TF_OD_API_MODEL_FILE = "burgerfries_toco.tflite";
private static final String TF_OD_API_LABELS_FILE = "file:///android_asset/burgerfries_labelmap.txt";
private static final DetectorMode MODE = DetectorMode.TF_OD_API;
// Minimum detection confidence to track a detection.
private static final float MINIMUM_CONFIDENCE_TF_OD_API = 0.5f;
private static final boolean MAINTAIN_ASPECT = false;
private static final Size DESIRED_PREVIEW_SIZE = new Size(width: 640, height: 480);
private static final boolean SAVE_PREVIEW_BITMAP = false;
private static final float TEXT_SIZE_DIP = 10;
OverlayView trackingOverlay;
private Integer sensorOrientation;
private Classifier detector;


Android
app
manifests
java
org.tensorflow.lite.examples.detection
customview
env
tflite
tracking
CameraActivity
CameraConnectionFragment
DetectorActivity
LegacyCameraConnectionFragment
java (generated)
assets
burgerfries_labelmap.txt
burgerfries_toco.tflite
detect_orig.tflite
labelmap_orig.txt
res
res (generated)
Gradle Scripts
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Version)
3
app
DetectorActivity.java
rectangle.xml
camera_connection_fragment.xml
Configure project in Project Structure dialog.
aaptOptions {
noCompress "tflite"
compileOptions {
sourceCompatibility = '1.8'
targetCompatibility = '1.8'
lintOptions {
abortOnError false
// import DownloadModels task
project.ext.ASSET_DIR = projectDir.toString() + '/src/main/assets'
project.ext.TMP_DIR = project.buildDir.toString() + '/downloads'
// Download default models; if you wish to use your own models then
// place them in the "assets" directory and comment out this line.
//apply from: "download_model.gradle"
//apply from:'download_model.gradle'
dependencies {
implementation fileTree(dir: 'libs', include: ['*.jar','*.aar'])
implementation 'androidx.appcompat:appcompat:1.0.0'
implementation 'androidx.coordinatorlayout:coordinatorlayout:1.0.0'
implementation 'com.google.android.material:material:1.0.0'
implementation 'org.tensorflow:tensorflow-lite:0.0.0-nightly'

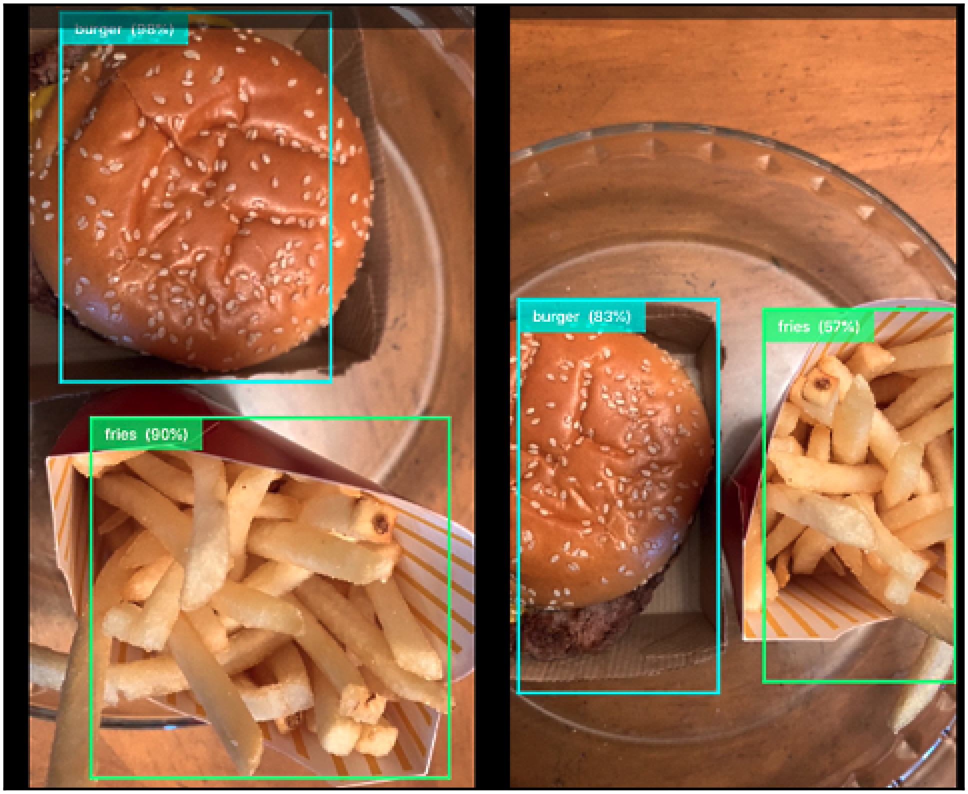

burger 98.05%
96%
09:44

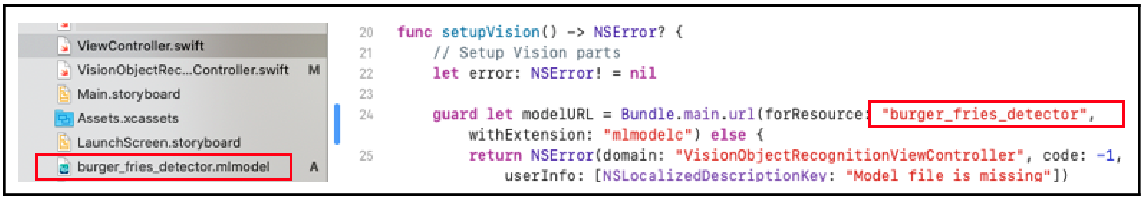

fries
0.95
burger
0.42

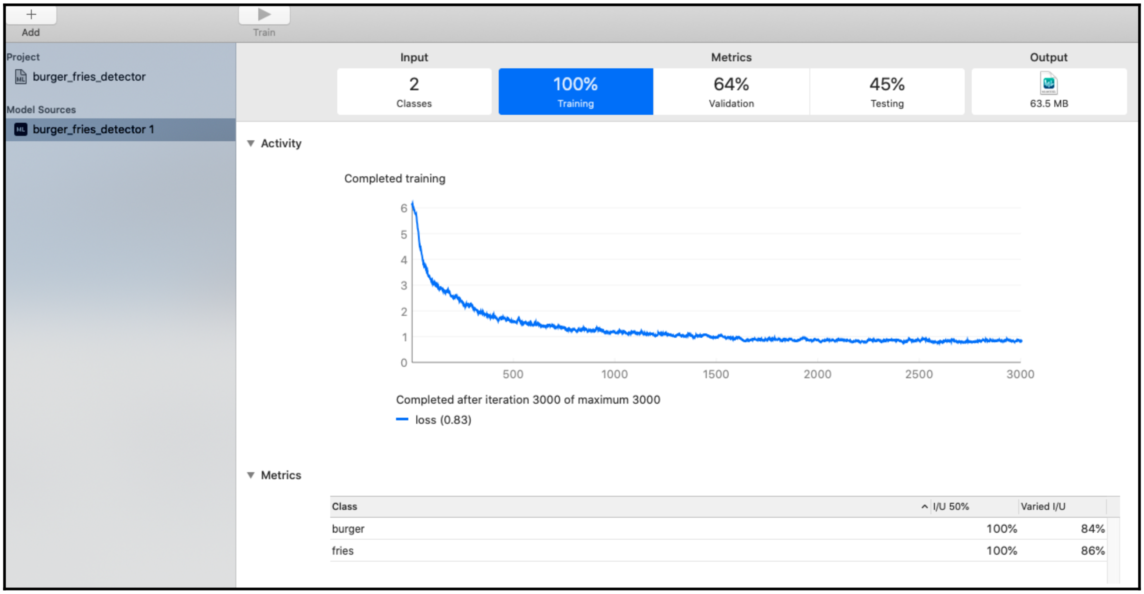
```
$ cd examples-master/lite/examples/object_detection/ios
$ pod install
Analyzing dependencies
Adding spec repo `trunk` with CDN `https://cdn.cocoapods.org/`

CocoaPods 1.9.0.beta.2 is available.
To update use: `sudo gem install cocoapods --pre`
[!] This is a test version we'd love you to try.

For more information, see https://blog.cocoapods.org and the CHANGELOG for this version at https:
//github.com/CocoaPods/CocoaPods/releases/tag/1.9.0.beta.2

Downloading dependencies
Installing TensorFlowLiteC (2.1.0)
Installing TensorFlowLiteSwift (2.1.0)
Generating Pods project
Integrating client project

[!] Please close any current Xcode sessions and use `ObjectDetection.xcworkspace` for this projec
t from now on.
Pod installation complete! There is 1 dependency from the Podfile and 2 total pods installed.
```



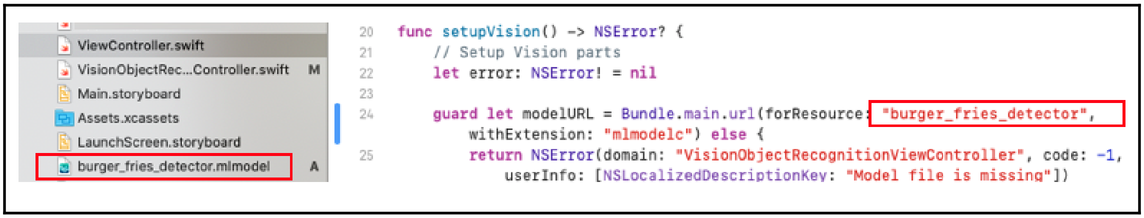

remote (56%)
cell phone (76%)


fries (90%)
burger (83%)
fries (57%)


ViewController.swift
VisionObjectRec...Controller.swift M
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
burger_fries_detector.mlmodel A
func setupVision() -> NSError? {
// Setup Vision parts
let error: NSError! = nil
guard let modelURL = Bundle.main.url(forResource: "burger_fries_detector",
withExtension: "mlmodelc") else {
return NSError(domain: "VisionObjectRecognitionViewController", code: -1,
userInfo: [NSLocalizedDescriptionKey: "Model file is missing"])

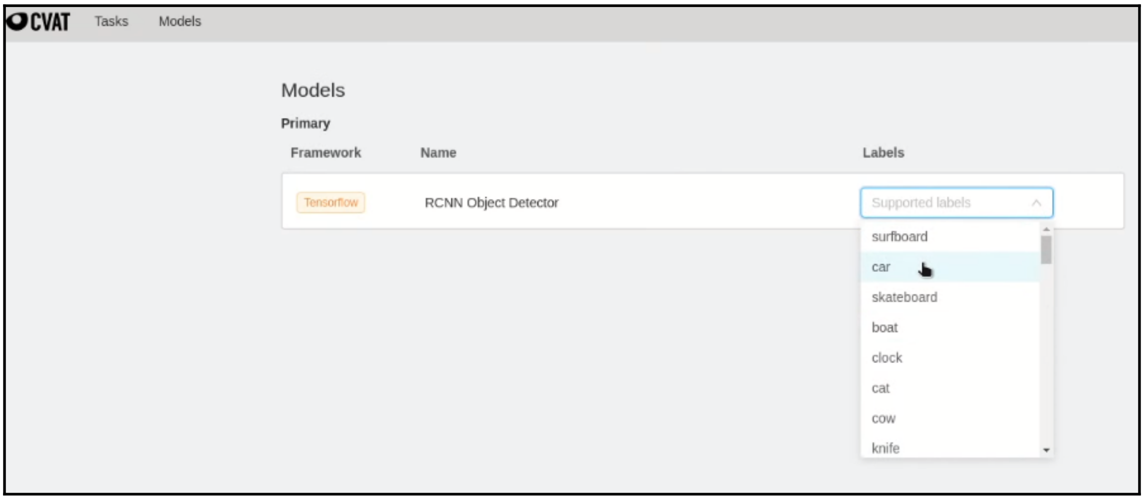


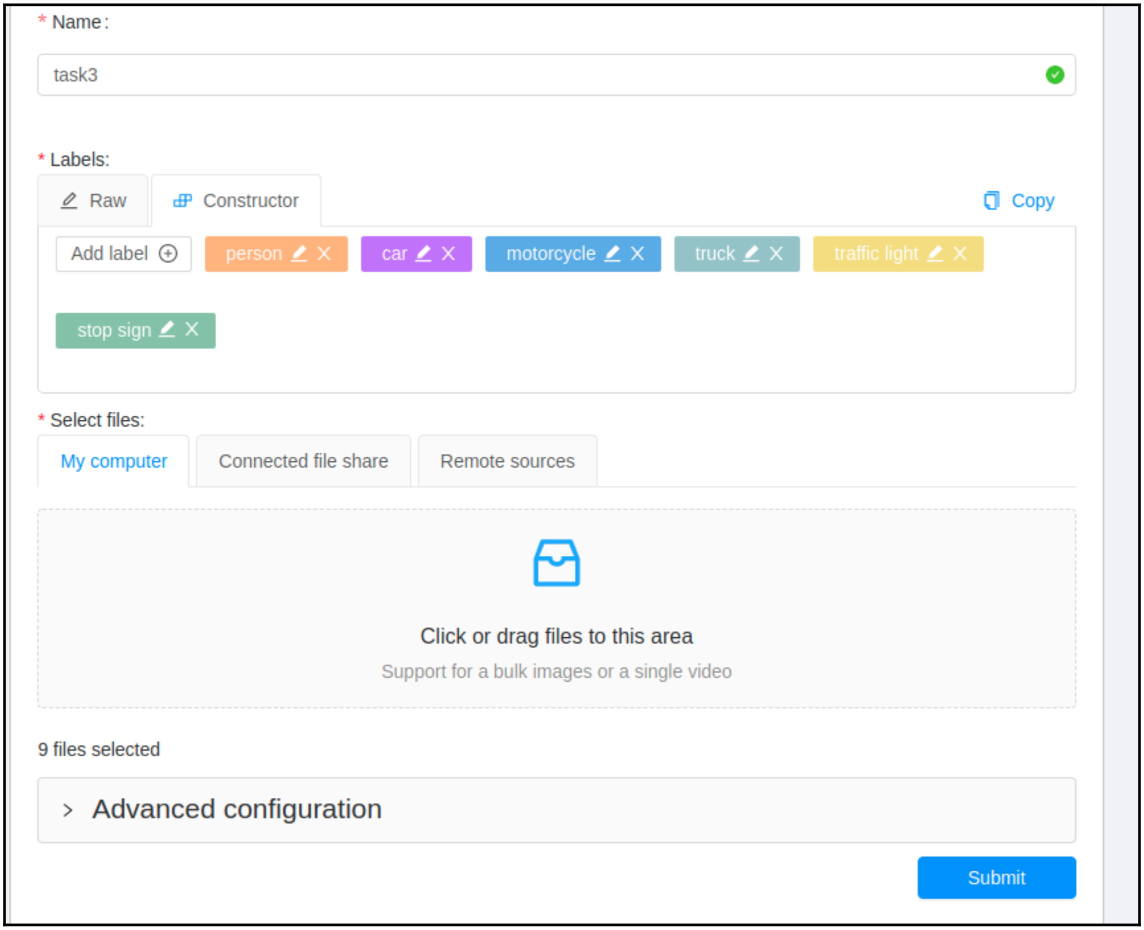
Add
Train
Project
burger_fries_detector
Model Sources
burger_fries_detector 1
Input
Metrics
Output
2
Classes
100%
Training
64%
Validation
45%
Testing
63.5 MB
Activity
Completed training
Completed after iteration 3000 of maximum 3000
loss (0.83)
Metrics
Class
I/U 50%
Varied I/U
burger
100%
84%
fries
100%
86%

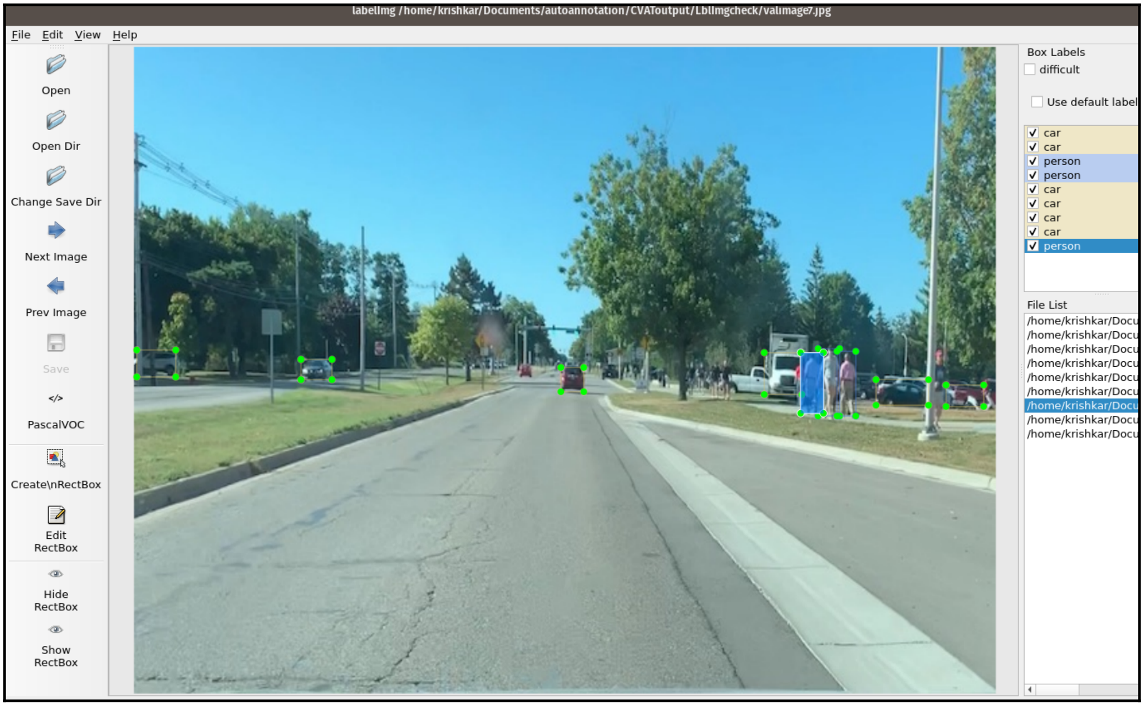


ViewController.swift
VisionObjectRec...Controller.swift
M
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
burger_fries_detector.mlmodel
A
func setupVision() -> NSError? {
// Setup Vision parts
let error: NSError! = nil
guard let modelURL = Bundle.main.url(forResource: "burger_fries_detector",
withExtension: "mlmodelc") else {
return NSError(domain: "VisionObjectRecognitionViewController", code: -1,
userInfo: [NSLocalizedDescriptionKey: "Model file is missing"])

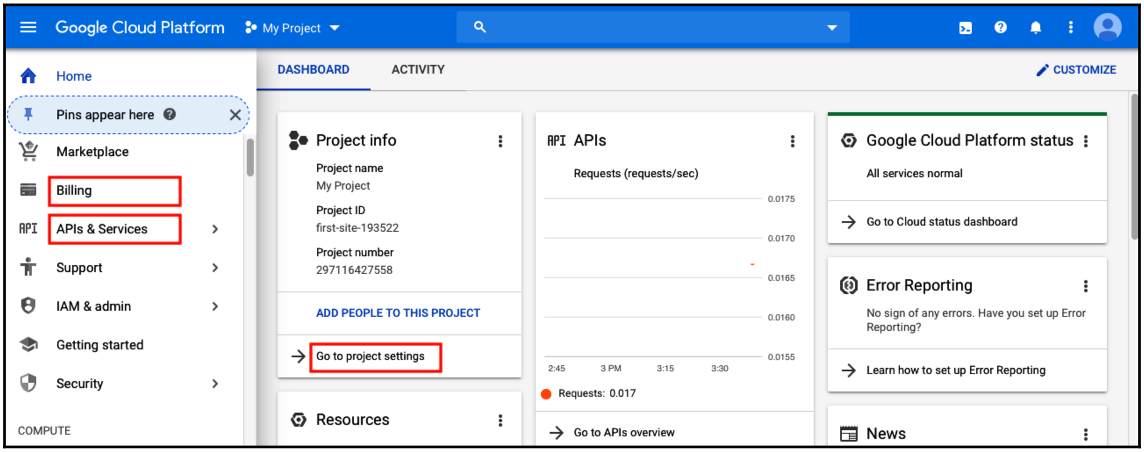

burger
Confidence: 1.00
fries
Confidence: 1.00

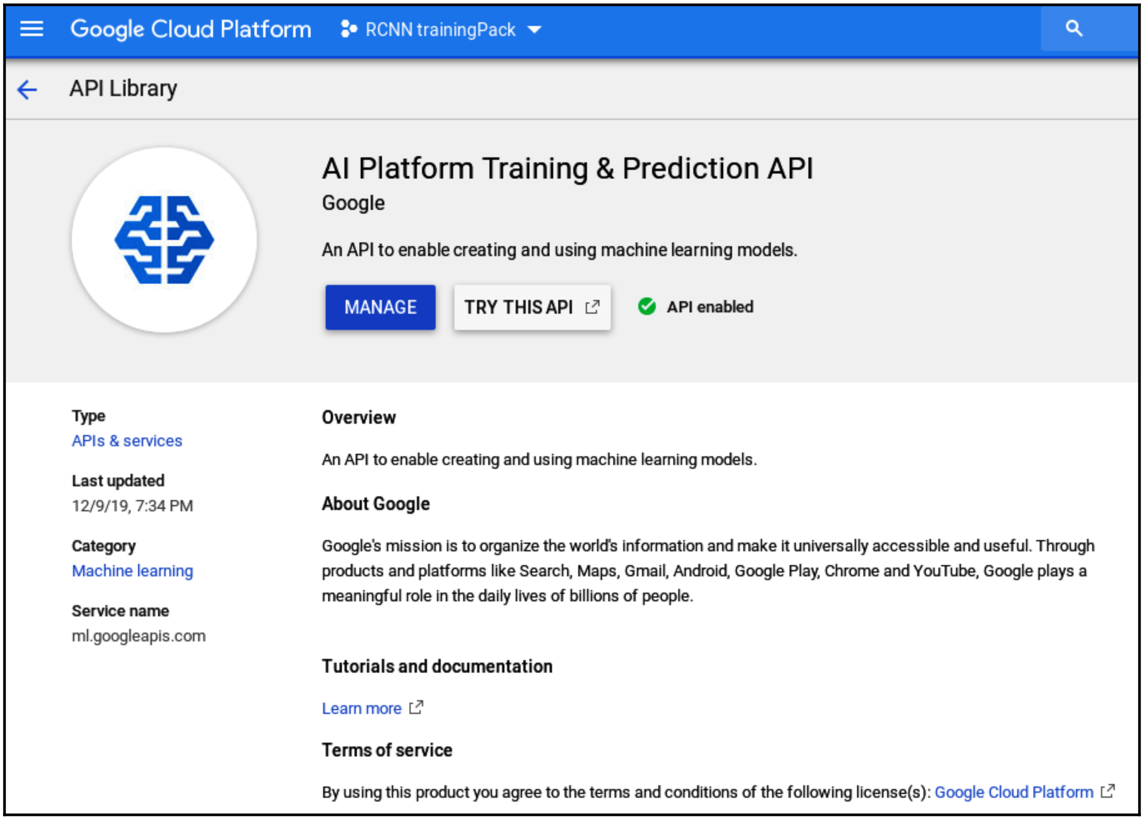

CVAT
Tasks
Models
Models
Primary
Framework
Name
Labels
Tensorflow
RCNN Object Detector
Supported labels
surfboard
car
skateboard
boat
clock
cat
cow
knife

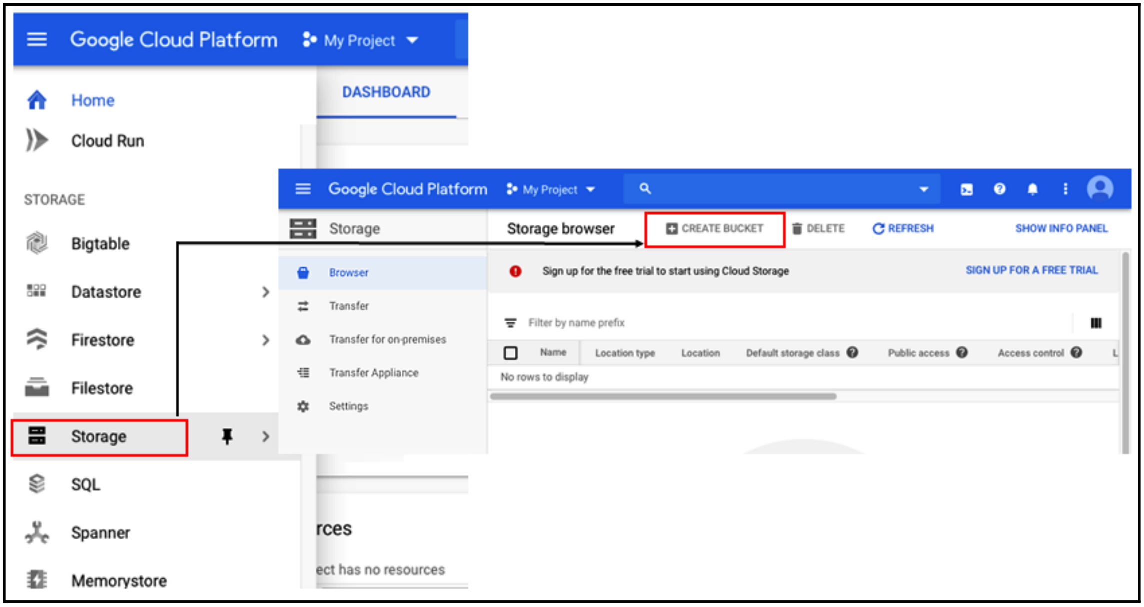

* Name:
task3
* Labels:
Raw
Constructor
Copy
Add label
person
car
motorcycle
truck
traffic light
stop sign
* Select files:
My computer
Connected file share
Remote sources
Click or drag files to this area
Support for a bulk images or a single video
9 files selected
Advanced configuration
Submit


labelImg /home/krishkar/Documents/autoannotation/CVAToutput/LblImgcheck/valimage7.jpg
File
Edit
View
Help
Open
Open Dir
Change Save Dir
Next Image
Prev Image
Save
</>
PascalVOC
Create\nRectBox
Edit RectBox
Hide RectBox
Show RectBox
Box Labels
difficult
Use default label
car
car
person
person
car
car
car
car
person
File List
/home/krishkar/Docu
/home/krishkar/Docu
/home/krishkar/Docu
/home/krishkar/Docu
/home/krishkar/Docu
/home/krishkar/Docu
/home/krishkar/Docu
/home/krishkar/Docu
/home/krishkar/Docu

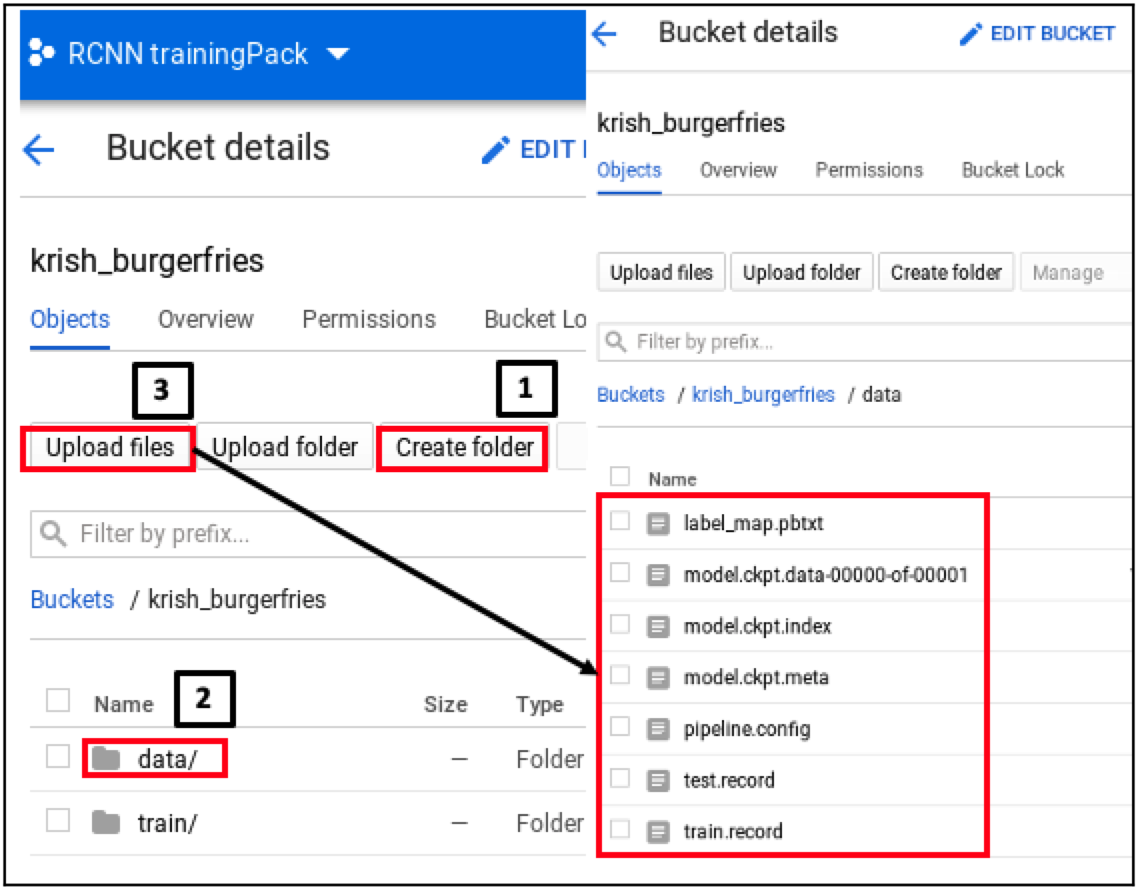
Chapter 12: Cloud Computing Platform for Computer Vision



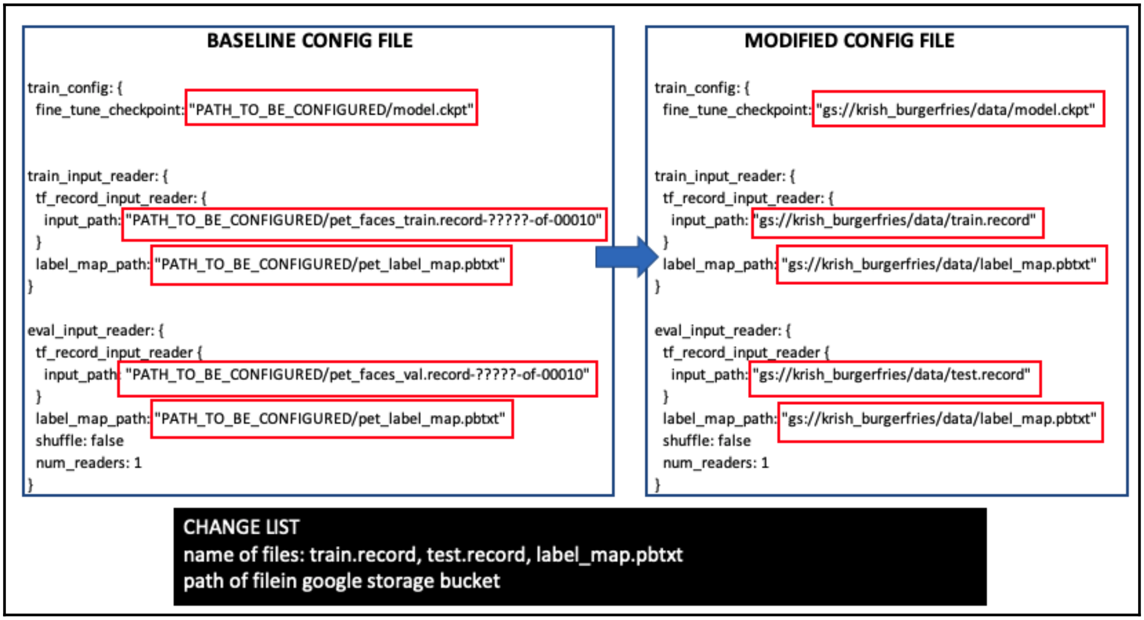

Google Cloud Platform
RCNN trainingPack
API Library
AI Platform Training & Prediction API
Google
An API to enable creating and using machine learning models.
MANAGE
TRY THIS API
API enabled
Type
APIs & services
Last updated
12/9/19, 7:34 PM
Category
Machine learning
Service name
ml.googleapis.com
Overview
An API to enable creating and using machine learning models.
About Google
Google's mission is to organize the world's information and make it universally accessible and useful. Through products and platforms like Search, Maps, Gmail, Android, Google Play, Chrome and YouTube, Google plays a meaningful role in the daily lives of billions of people.
Tutorials and documentation
Learn more
Terms of service
By using this product you agree to the terms and conditions of the following license(s): Google Cloud Platform

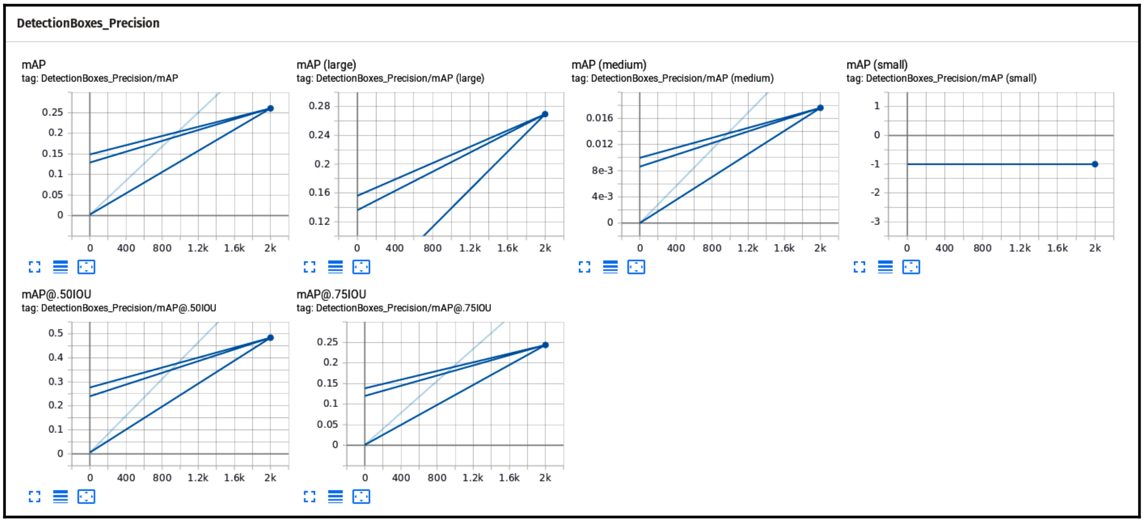


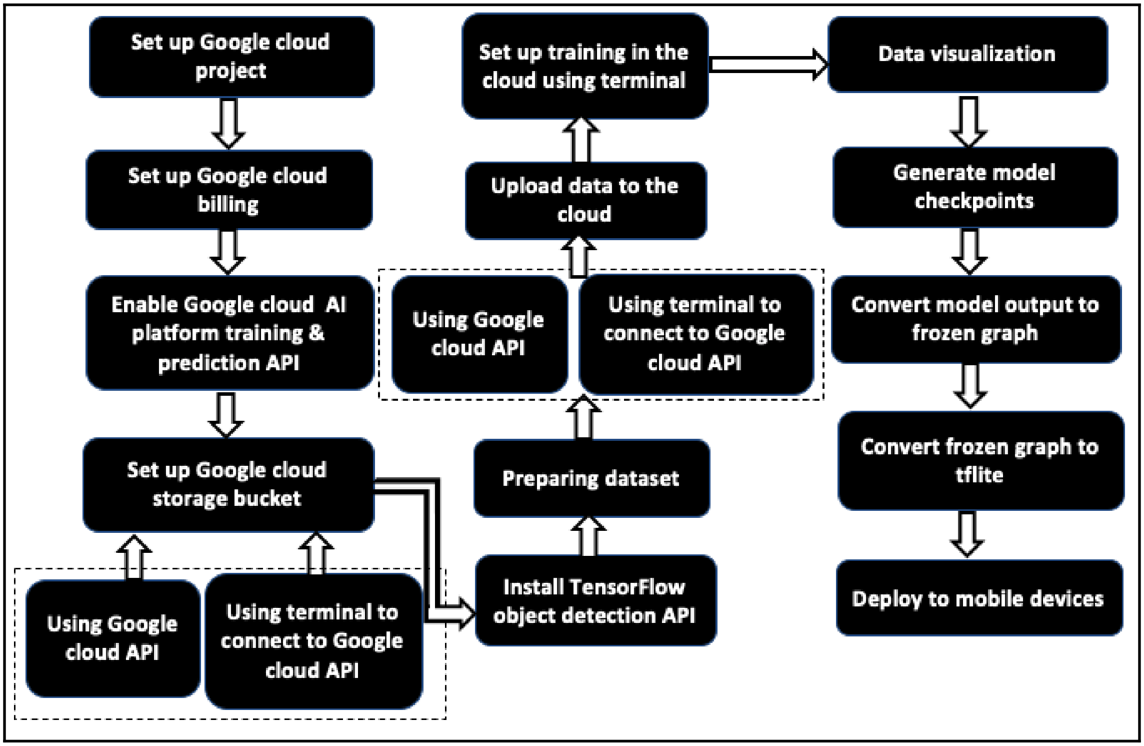
Google Cloud Platform
My Project
DASHBOARD
Home
Cloud Run
STORAGE
Bigtable
Datastore
Firestore
Filestore
Storage
SQL
Spanner
Memorystore
rces
ect has no resources
Google Cloud Platform
My Project
Storage
Storage browser
CREATE BUCKET
DELETE
REFRESH
SHOW INFO PANEL
Browser
Transfer
Transfer for on-premises
Transfer Appliance
Settings
Sign up for the free trial to start using Cloud Storage
SIGN UP FOR A FREE TRIAL
Filter by name prefix
Name
Location type
Location
Default storage class
Public access
Access control
No rows to display

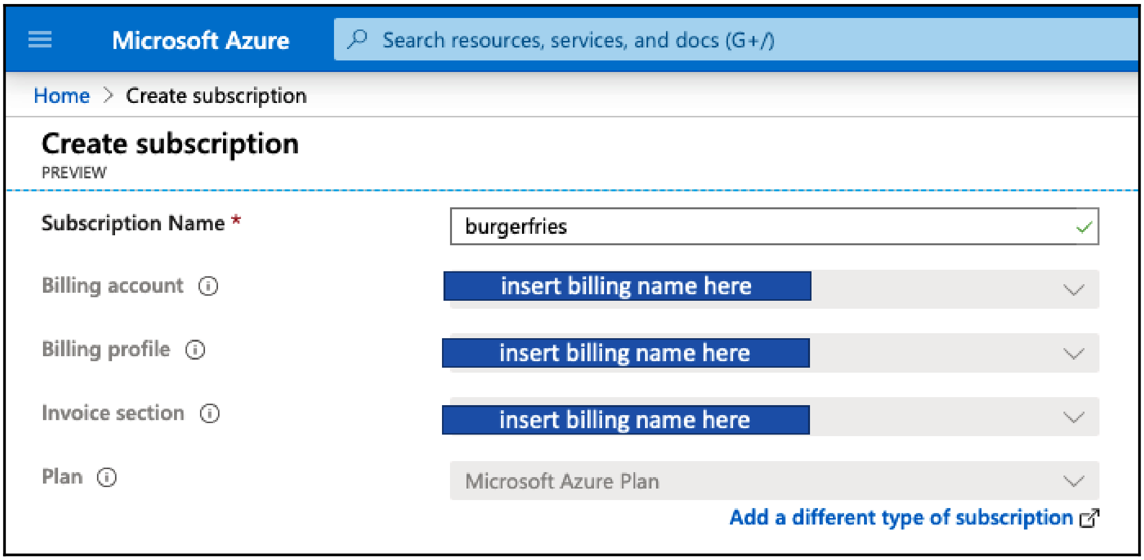


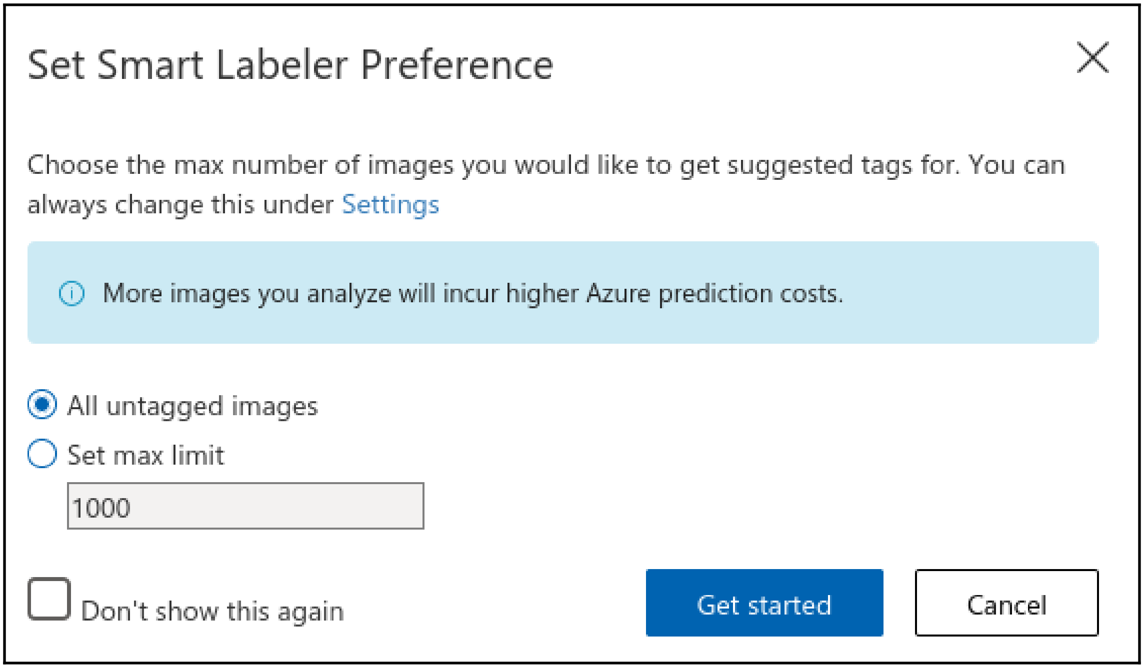
RCNN trainingPack
Bucket details
EDIT
krish_burgerfries
Objects
Overview
Permissions
Bucket Lo
3
1
Upload files
Upload folder
Create folder
Filter by prefix...
Buckets / krish_burgerfries
Name
2
Size
Type
data/
Folder
train/
Folder
Bucket details
EDIT BUCKET
krish_burgerfries
Objects
Overview
Permissions
Bucket Lock
Upload files
Upload folder
Create folder
Manage
Filter by prefix...
Buckets / krish_burgerfries / data
Name
label_map.pbtxt
model.ckpt.data-00000-of-00001
model.ckpt.index
model.ckpt.meta
pipeline.config
test.record
train.record

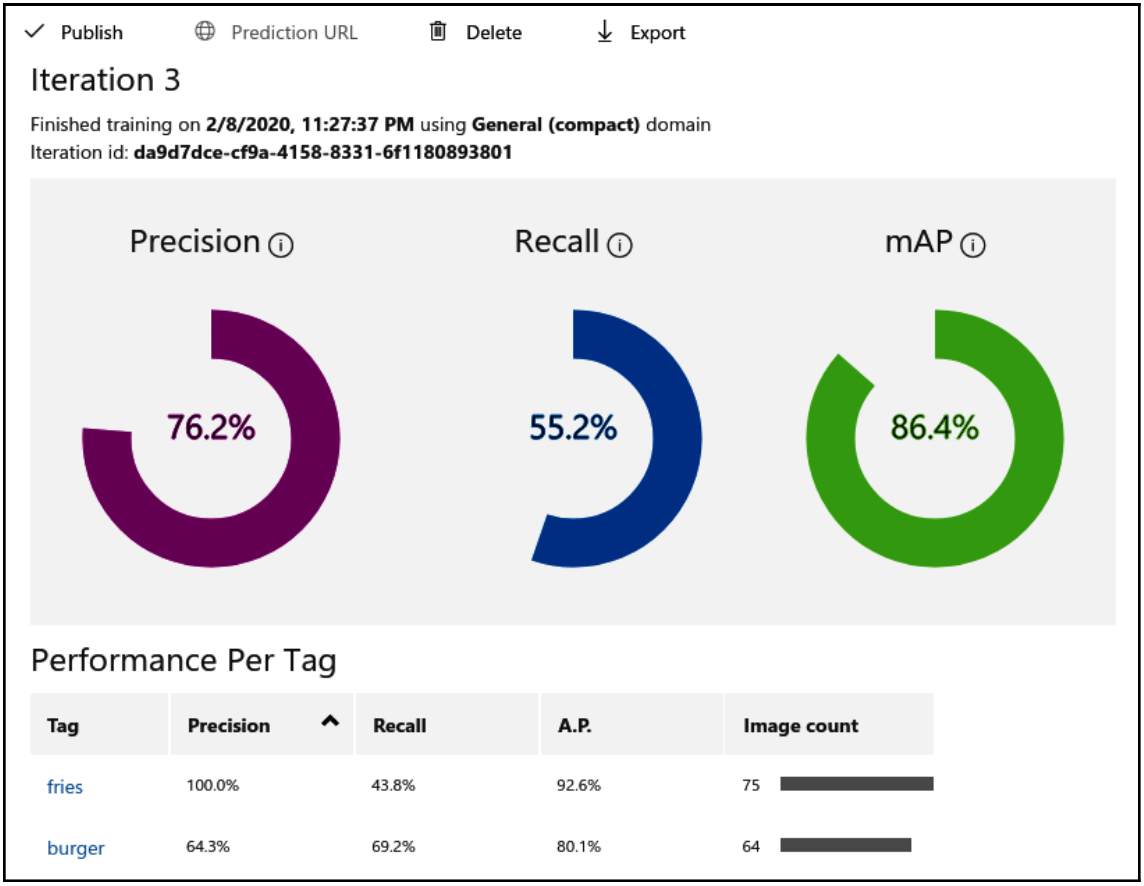

BASELINE CONFIG FILE
train_config: {
fine_tune_checkpoint: "PATH_TO_BE_CONFIGURED/model.ckpt"
train_input_reader: {
tf_record_input_reader: {
input_path: "PATH_TO_BE_CONFIGURED/pet_faces_train.record-?????-of-00010"
}
label_map_path: "PATH_TO_BE_CONFIGURED/pet_label_map.pbtxt"
}
eval_input_reader: {
tf_record_input_reader {
input_path: "PATH_TO_BE_CONFIGURED/pet_faces_val.record-?????-of-00010"
}
label_map_path: "PATH_TO_BE_CONFIGURED/pet_label_map.pbtxt"
shuffle: false
num_readers: 1
}
MODIFIED CONFIG FILE
train_config: {
fine_tune_checkpoint: "gs://krish_burgerfries/data/model.ckpt"
train_input_reader: {
tf_record_input_reader: {
input_path: "gs://krish_burgerfries/data/train.record"
}
label_map_path: "gs://krish_burgerfries/data/label_map.pbtxt"
}
eval_input_reader: {
tf_record_input_reader {
input_path: "gs://krish_burgerfries/data/test.record"
}
label_map_path: "gs://krish_burgerfries/data/label_map.pbtxt"
shuffle: false
num_readers: 1
}
CHANGE LIST
name of files: train.record, test.record, label_map.pbtxt
path of filein google storage bucket

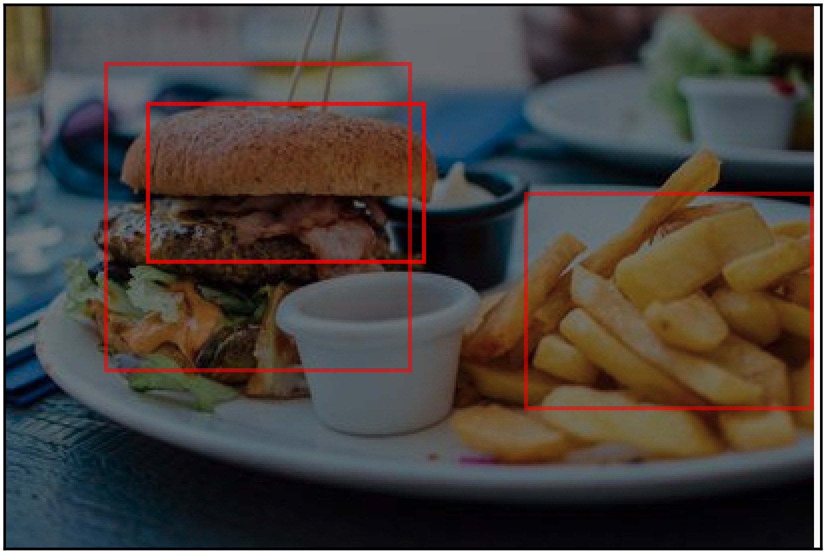

DetectionBoxes_Precision
mAP
tag: DetectionBoxes_Precision/mAP
mAP (large)
tag: DetectionBoxes_Precision/mAP (large)
mAP (medium)
tag: DetectionBoxes_Precision/mAP (medium)
mAP (small)
tag: DetectionBoxes_Precision/mAP (small)
mAP@.50IOU
tag: DetectionBoxes_Precision/mAP@.50IOU
mAP@.75IOU
tag: DetectionBoxes_Precision/mAP@.75IOU
0
400
800
1.2k
1.6k
2k

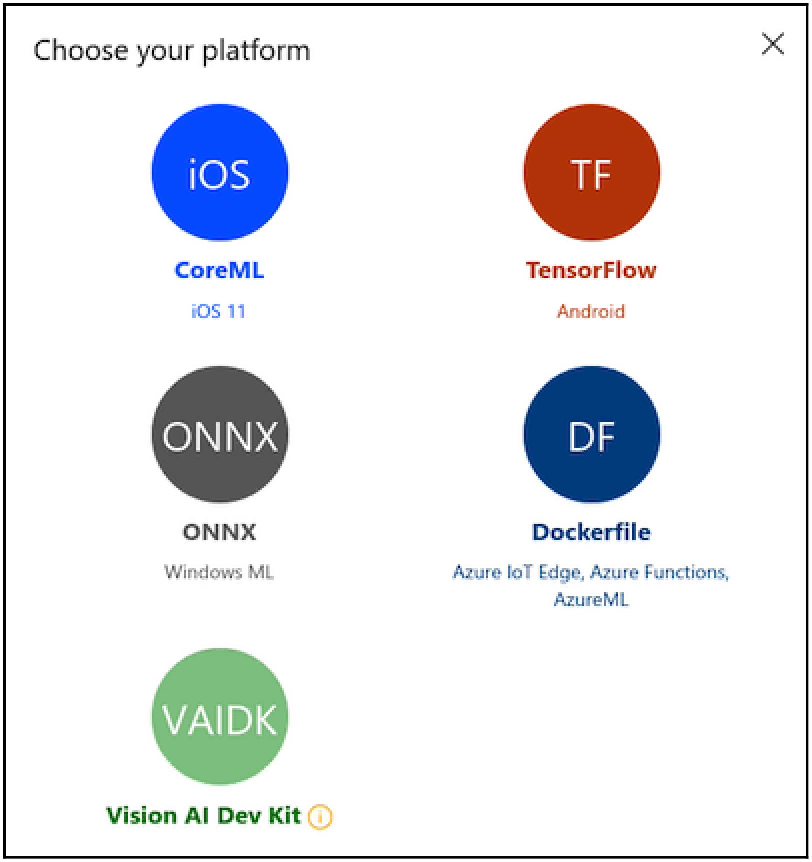

Set up Google cloud project
Set up Google cloud billing
Enable Google cloud AI platform training & prediction API
Set up Google cloud storage bucket
Using Google cloud API
Using terminal to connect to Google cloud API
Install TensorFlow object detection API
Preparing dataset
Using Google cloud API
Using terminal to connect to Google cloud API
Upload data to the cloud
Set up training in the cloud using terminal
Data visualization
Generate model checkpoints
Convert model output to frozen graph
Convert frozen graph to tflite
Deploy to mobile devices


Microsoft Azure
Search resources, services, and docs (G+/)
Home > Create subscription
Create subscription
PREVIEW
Subscription Name *
burgerfries
Billing account
insert billing name here
Billing profile
insert billing name here
Invoice section
insert billing name here
Plan
Microsoft Azure Plan
Add a different type of subscription

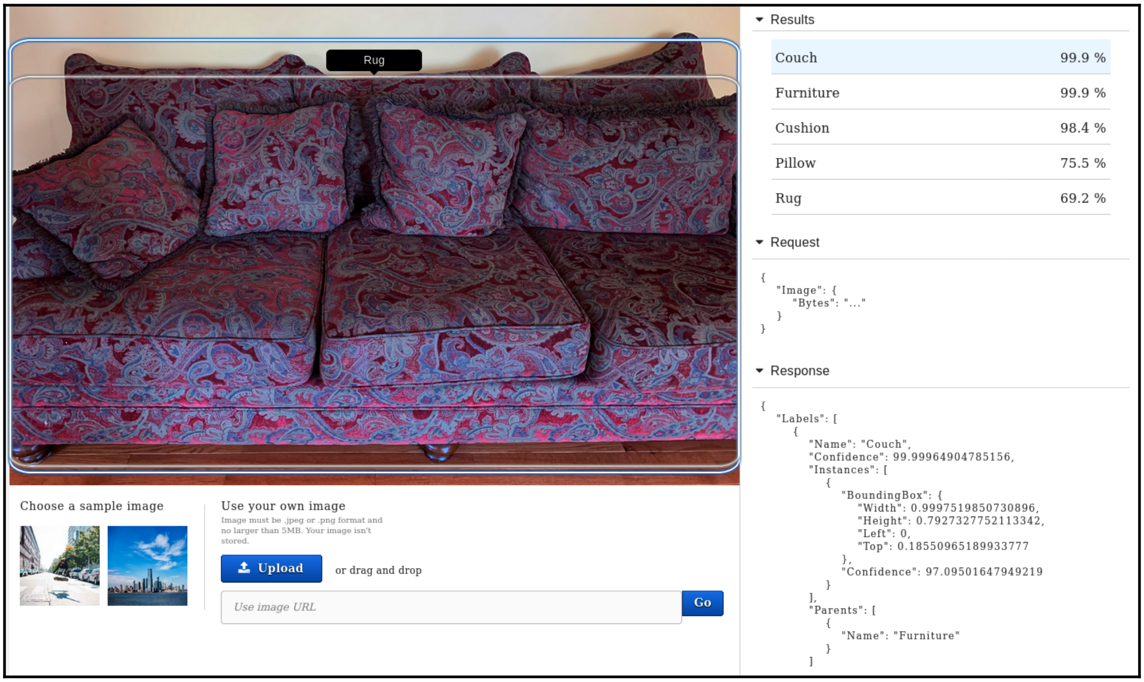

Set Smart Labeler Preference
Choose the max number of images you would like to get suggested tags for. You can always change this under Settings
More images you analyze will incur higher Azure prediction costs.
All untagged images
Set max limit
1000
Don't show this again
Get started
Cancel

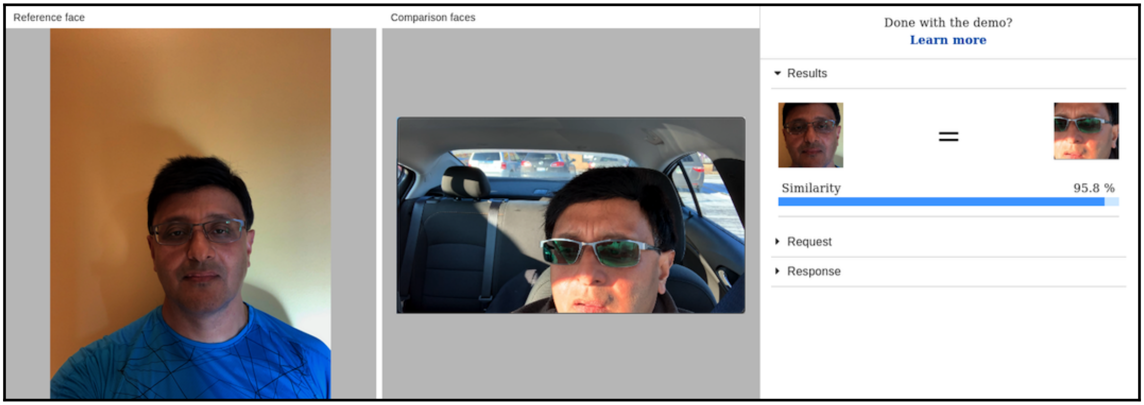

Publish
Prediction URL
Delete
Export
Iteration 3
Finished training on 2/8/2020, 11:27:37 PM using General (compact) domain
Iteration id: da9d7dce-cf9a-4158-8331-6f1180893801
Precision
76.2%
Recall
55.2%
mAP
86.4%
Performance Per Tag
Tag
Precision
Recall
A.P.
Image count
fries
100.0%
43.8%
92.6%
75
burger
64.3%
69.2%
80.1%
64




Choose your platform
iOS
CoreML
iOS 11
TF
TensorFlow
Android
ONNX
ONNX
Windows ML
DF
Dockerfile
Azure IoT Edge, Azure Functions, AzureML
VAIDK
Vision AI Dev Kit


Objects
Labels
Web
Text
Properties
Safe Search
Couch
98%
sofa.jpeg


Rug
Results
Couch 99.9 %
Furniture 99.9 %
Cushion 98.4 %
Pillow 75.5 %
Rug 69.2 %
Request
{ "Image": { "Bytes": "..." } }
Response
{ "Labels": [{ "Name": "Couch", "Confidence": 99.99964904785156, "Instances": [{ "BoundingBox": { "Width": 0.9997519850730896, "Height": 0.7927327752113342, "Left": 0, "Top": 0.18550965189933777 }, "Confidence": 97.09501647949219 }], "Parents": [{ "Name": "Furniture" }]
Choose a sample image
Use your own image
Image must be .jpeg or .png format and no larger than 5MB. Your image isn't stored.
Upload
or drag and drop
Use image URL
Go


Reference face
Comparison faces
Done with the demo?
Learn more
Results
Similarity
95.8 %
Request
Response