# CNN Car Detection

February 20, 2022

IBM Machine Learning Professional Certificate **Deep Learning and Reinforcement Learning**

## 1 CNN on Car Detection (VGG vs LeNet)

---

**Author**: Chawit Kaewnuratchadasorn **Date**: 18th Feb 2022

This notebook was created for Deep Learning and Unsupervised Learning of IBM Machine Learning certificate. The dataset was obtained from Kaggle Dataset by Edward Zhang. The link was attached below. In this notebook, the VGG and LeNet convolutional neural networks are applied to detect cars in an image. This project aims to learn Region-Based Convolutional Neural Network and learn tensorflow model generation.

Data source: Car Detection Dataset

The contents include: > 1. Overview of Dataset > 2. Region-based Processing > 3. Convolutoinal Neural Networks » VGG » LeNet > 4. Summary and Future Plan

Firstly, below are all dependencies of this notebook.

[1]:
```
# General tools for processing
import os
import numpy as np
import cv2
import pandas as pd

# visualisation tools
import visualkeras
from matplotlib import pyplot as plt
%matplotlib inline

import warnings
warnings.filterwarnings("ignore")

# Tools for preparing
from sklearn.model_selection import train_test_split

# Tools for neural networks
import tensorflow as tf
import keras
from keras.datasets import cifar10
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Conv2D, MaxPooling2D, AveragePooling2D
from keras.losses import categorical_crossentropy
```

```
Using TensorFlow backend.
```

## 1.1 Overview of Dataset

The dataset has 559 rows and 5 columns, including image file name and rectangular mark of car by two diagonal points. Each image is in another folder. The dataset is in tablular form as seen below.

[2]:

```
data = pd.read_csv("data/train_solution_bounding.csv")
data.head()
```

[2]:

```
              image        xmin        ymin        xmax        ymax
0   vid_4_1000.jpg  281.259045  187.035071  327.727931  223.225547
1  vid_4_10000.jpg   15.163531  187.035071  120.329957  236.430180
2  vid_4_10040.jpg  239.192475  176.764801  361.968162  236.430180
3  vid_4_10020.jpg  496.483358  172.363256  630.020260  231.539575
4  vid_4_10060.jpg   16.630970  186.546010  132.558611  238.386422
```

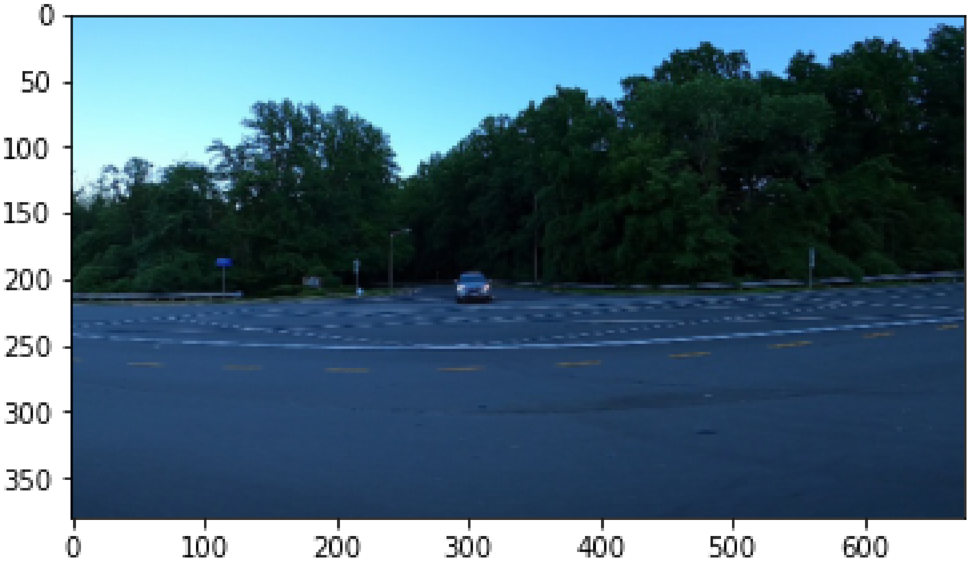
This is the image in the first row of table. The red rectangular mark is drawn from the xmin, ymin, xmax, and ymax on the table.

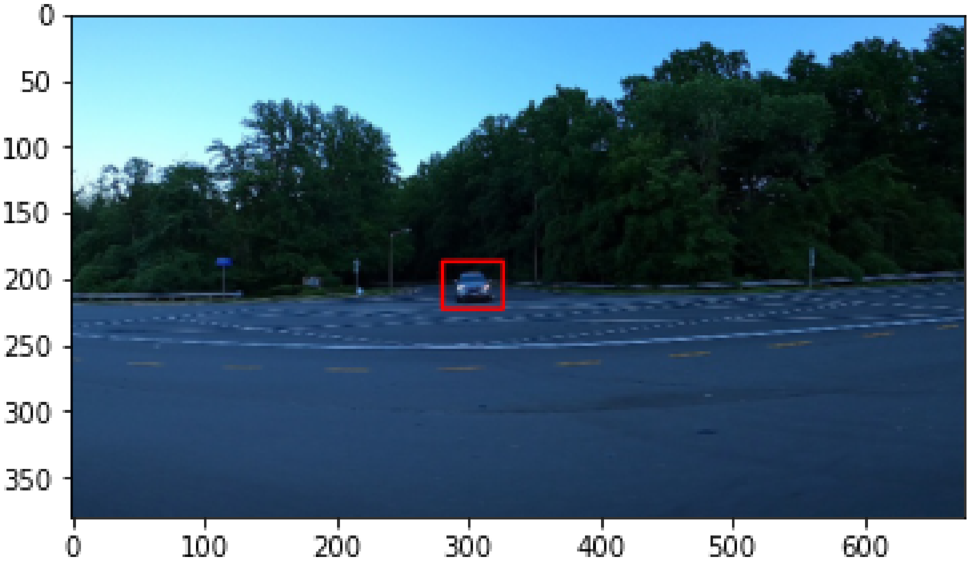
[3]:

```
x = data[data["image"] == "vid_4_1000.jpg"]

photo = plt.imread(x["image"][0])
plt.imshow(photo)

pt1=(int(x['xmin'][0]),int(x['ymin'][0]))
pt2=(int(x['xmax'][0]),int(x['ymax'][0]))

color=(255, 0, 0)
thickness = 2
cv2.rectangle(photo, pt1, pt2, color, thickness)
plt.figure()
plt.imshow(photo)
```

[3]:

```
<matplotlib.image.AxesImage at 0x7f8e0b848dd0>
```





[4]:
```
all_photos = []

os.chdir("data/training_images")

for i in data.index:
    photo = plt.imread(data["image"][i])
    all_photos.append(photo)
```

```
os.chdir("../../")
```

## 1.2 Region-based Processing

Here is the idea of object detection in this notebook. Each image file is segmented in to many images by selective search segmentation of OpenCV library. In this section, segmentation and intersection over union will be explained before combining together. The methods were obtained from this link

Selective Search Segmentation Intersect Over Union (IOU)

### 1.2.1 Selective Search Segmentation

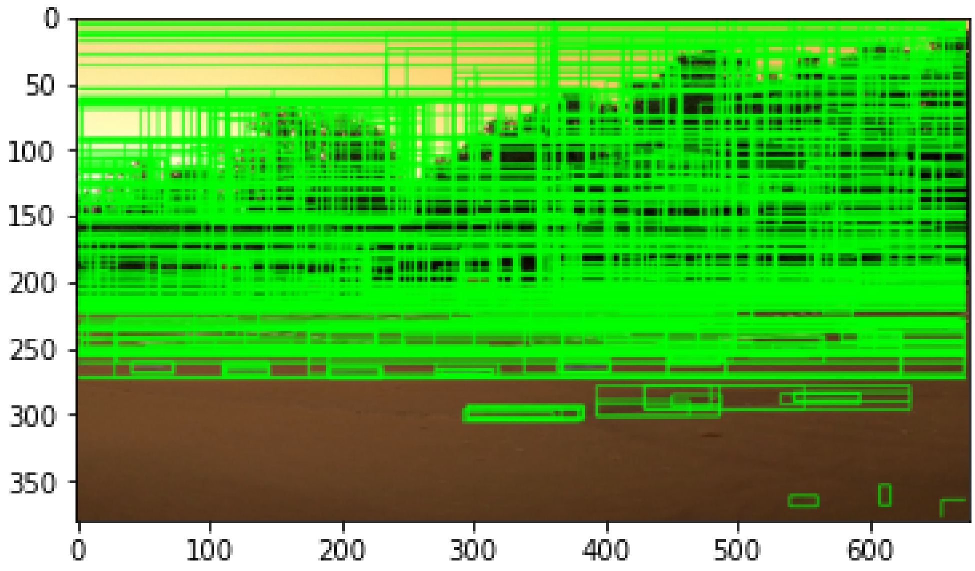
Each image is segmented as seen below. The green blocks show the segmentation of image number 1000.

[5]:

```
cv2.setUseOptimized(True);
ss = cv2.ximgproc.segmentation.createSelectiveSearchSegmentation()

path = "data/training_images"

im = cv2.imread(os.path.join(path,"vid_4_1000.jpg"))
ss.setBaseImage(im)
ss.switchToSelectiveSearchFast()
rects = ss.process()
imOut = im.copy()

for i, rect in (enumerate(rects)):
    x, y, w, h = rect
    cv2.rectangle(imOut, (x, y), (x+w, y+h), (0, 255, 0), 1, cv2.LINE_AA)

plt.imshow(imOut)
```

[5]:

```
<matplotlib.image.AxesImage at 0x7f8dd445c610>
```



### 1.2.2 Intersect Over Union

Each image is calculated for the intersection area. The reference area is the area of xmin, ymin, xmax, and ymax which were obtained from the table. Then, each intersection area is calculated for each segment from selective search segmentation.

[6]:

```
def get_iou(bb1, bb2):

    assert bb1['x1'] < bb1['x2'] #bb1
    assert bb1['y1'] < bb1['y2']

    assert bb2['x1'] < bb2['x2'] #bb2
    assert bb2['y1'] < bb2['y2'];

    x_left = max(bb1['x1'], bb2['x1'])
    y_top = max(bb1['y1'], bb2['y1'])
    x_right = min(bb1['x2'], bb2['x2'])
    y_bottom = min(bb1['y2'], bb2['y2'])

    if x_right < x_left or y_bottom < y_top:
        return 0.0

    intersection_area = (x_right - x_left) * (y_bottom - y_top)
    bb1_area = (bb1['x2'] - bb1['x1']) * (bb1['y2'] - bb1['y1'])
    bb2_area = (bb2['x2'] - bb2['x1']) * (bb2['y2'] - bb2['y1'])
    iou = intersection_area / float(bb1_area + bb2_area - intersection_area)

    assert iou >= 0.0
    assert iou <= 1.0

    return iou
```

Here, this is the combination of selective search segmentation and intersect of union. The image has also been reshaped into 224x224.

[7]:

```
image_liste=[]
k=0
l=0
z=0 #Loading
for a in pd.read_csv('data/train_solution_bounding.csv').values:
    Name,xmin,ymin,xmax,ymax=a
    bb1={'x1':int(xmin),'y1':int(ymin),'x2':int(xmax),'y2':int(ymax)}
    try:
        img=cv2.imread('data/training_images/'+Name)
        ss.setBaseImage(img)
        ss.switchToSelectiveSearchFast()
        rects = ss.process()
        for i in rects:
            x, y, w, h = i # Selective bounty boxxes
            bb2={'x1':x,'y1':y,'x2':x+w,'y2':y+h}
            img1=img[bb2['y1']:bb2['y2'],bb2['x1']:bb2['x2']] # Crop img
            img1_shape=cv2.resize(img1,(224,224))
            if k<l:
                if 0.5<get_iou(bb1,bb2):
                    image_liste.append([img1_shape,1])
                    k+=1
            else:
                if 0.5<get_iou(bb1,bb2):
                    image_liste.append([img1_shape,1])
                    k+=1
                else:
                    image_liste.append([img1_shape,0])
                    l+=1
    except Exception as e:
        print('Error',e)
    z+=1
    print(Name,z,len(rects))
```

```
vid_4_1000.jpg 1 1133
vid_4_10000.jpg 2 1426
vid_4_10040.jpg 3 1446
vid_4_10020.jpg 4 1402
vid_4_10060.jpg 5 1475
vid_4_10100.jpg 6 1712
vid_4_10120.jpg 7 1531
vid_4_10140.jpg 8 1630
```

```
vid_4_1020.jpg 9 1120
vid_4_1040.jpg 10 1220
vid_4_10480.jpg 11 1714
vid_4_10500.jpg 12 1771
vid_4_10520.jpg 13 1724
vid_4_1060.jpg 14 1323
vid_4_10960.jpg 15 1865
vid_4_10980.jpg 16 1816
vid_4_11000.jpg 17 1770
vid_4_11020.jpg 18 1657
vid_4_11240.jpg 19 1725
vid_4_11260.jpg 20 1912
vid_4_11280.jpg 21 1969
vid_4_11380.jpg 22 1779
vid_4_11400.jpg 23 1851
vid_4_11420.jpg 24 1904
vid_4_11440.jpg 25 1842
vid_4_11900.jpg 26 1973
vid_4_11880.jpg 27 1881
vid_4_11920.jpg 28 1935
vid_4_11940.jpg 29 1917
vid_4_11960.jpg 30 2034
vid_4_11980.jpg 31 2081
vid_4_12000.jpg 32 1848
vid_4_12040.jpg 33 2052
vid_4_12100.jpg 34 1917
vid_4_12060.jpg 35 2009
vid_4_12080.jpg 36 2034
vid_4_12120.jpg 37 1863
vid_4_12140.jpg 38 1665
vid_4_12140.jpg 39 1665
vid_4_12160.jpg 40 1697
vid_4_12160.jpg 41 1697
vid_4_12160.jpg 42 1697
vid_4_12180.jpg 43 1599
vid_4_12180.jpg 44 1599
vid_4_12180.jpg 45 1599
vid_4_12200.jpg 46 1399
vid_4_12200.jpg 47 1399
vid_4_12220.jpg 48 1479
vid_4_12240.jpg 49 1483
vid_4_12240.jpg 50 1483
vid_4_12260.jpg 51 1963
vid_4_12260.jpg 52 1963
vid_4_12280.jpg 53 2099
vid_4_12280.jpg 54 2099
vid_4_12300.jpg 55 2079
vid_4_12300.jpg 56 2079
```

```
vid_4_12300.jpg 57 2079
vid_4_12300.jpg 58 2079
vid_4_12320.jpg 59 1961
vid_4_12320.jpg 60 1961
vid_4_12320.jpg 61 1961
vid_4_12340.jpg 62 1904
vid_4_12340.jpg 63 1904
vid_4_12340.jpg 64 1904
vid_4_12360.jpg 65 1914
vid_4_12360.jpg 66 1914
vid_4_12380.jpg 67 1945
vid_4_12480.jpg 68 2043
vid_4_13580.jpg 69 1833
vid_4_13620.jpg 70 1981
vid_4_13640.jpg 71 2000
vid_4_13640.jpg 72 2000
vid_4_13660.jpg 73 2193
vid_4_13660.jpg 74 2193
vid_4_13680.jpg 75 2192
vid_4_13680.jpg 76 2192
vid_4_13680.jpg 77 2192
vid_4_13700.jpg 78 2305
vid_4_13700.jpg 79 2305
vid_4_13700.jpg 80 2305
vid_4_13700.jpg 81 2305
vid_4_13720.jpg 82 2107
vid_4_13720.jpg 83 2107
vid_4_13720.jpg 84 2107
vid_4_13740.jpg 85 2135
vid_4_13740.jpg 86 2135
vid_4_13760.jpg 87 2246
vid_4_13760.jpg 88 2246
vid_4_13760.jpg 89 2246
vid_4_13780.jpg 90 2297
vid_4_13780.jpg 91 2297
vid_4_13800.jpg 92 2199
vid_4_13800.jpg 93 2199
vid_4_13840.jpg 94 2303
vid_4_13840.jpg 95 2303
vid_4_13820.jpg 96 2265
vid_4_13820.jpg 97 2265
vid_4_13820.jpg 98 2265
vid_4_13860.jpg 99 2392
vid_4_13860.jpg 100 2392
vid_4_13860.jpg 101 2392
vid_4_13900.jpg 102 2250
vid_4_13880.jpg 103 2451
vid_4_13880.jpg 104 2451
```

```
vid_4_13920.jpg 105 2416
vid_4_14140.jpg 106 2160
vid_4_14160.jpg 107 1787
vid_4_14360.jpg 108 1561
vid_4_14340.jpg 109 1403
vid_4_14380.jpg 110 1532
vid_4_14400.jpg 111 1634
vid_4_14460.jpg 112 1247
vid_4_14440.jpg 113 1455
vid_4_14480.jpg 114 1173
vid_4_14500.jpg 115 1333
vid_4_15000.jpg 116 1832
vid_4_15020.jpg 117 1736
vid_4_15040.jpg 118 1843
vid_4_16000.jpg 119 1406
vid_4_16020.jpg 120 1369
vid_4_16040.jpg 121 1428
vid_4_16060.jpg 122 1359
vid_4_16100.jpg 123 1293
vid_4_16120.jpg 124 1410
vid_4_16140.jpg 125 1450
vid_4_16160.jpg 126 1383
vid_4_16180.jpg 127 1284
vid_4_16280.jpg 128 1450
vid_4_16320.jpg 129 1341
vid_4_16300.jpg 130 1517
vid_4_16400.jpg 131 1148
vid_4_16420.jpg 132 1196
vid_4_16500.jpg 133 1358
vid_4_16660.jpg 134 1526
vid_4_16680.jpg 135 1480
vid_4_16700.jpg 136 1619
vid_4_16720.jpg 137 1449
vid_4_17040.jpg 138 1571
vid_4_17060.jpg 139 1536
vid_4_17080.jpg 140 1581
vid_4_17100.jpg 141 1504
vid_4_17140.jpg 142 1486
vid_4_17140.jpg 143 1486
vid_4_17120.jpg 144 1433
vid_4_17160.jpg 145 1515
vid_4_17180.jpg 146 1513
vid_4_17240.jpg 147 1408
vid_4_17260.jpg 148 1249
vid_4_17320.jpg 149 1365
vid_4_17340.jpg 150 1474
vid_4_17340.jpg 151 1474
vid_4_17360.jpg 152 1370
```

```
vid_4_17400.jpg 153 1485
vid_4_17440.jpg 154 1501
vid_4_17420.jpg 155 1443
vid_4_17560.jpg 156 1679
vid_4_17560.jpg 157 1679
vid_4_17560.jpg 158 1679
vid_4_17540.jpg 159 1707
vid_4_17540.jpg 160 1707
vid_4_17580.jpg 161 1604
vid_4_17580.jpg 162 1604
vid_4_1760.jpg 163 1370
vid_4_17600.jpg 164 1789
vid_4_17620.jpg 165 1811
vid_4_17620.jpg 166 1811
vid_4_17640.jpg 167 1883
vid_4_17640.jpg 168 1883
vid_4_17660.jpg 169 1794
vid_4_17660.jpg 170 1794
vid_4_17680.jpg 171 1649
vid_4_1780.jpg 172 1500
vid_4_1780.jpg 173 1500
vid_4_1800.jpg 174 1613
vid_4_1800.jpg 175 1613
vid_4_1800.jpg 176 1613
vid_4_18180.jpg 177 1378
vid_4_1820.jpg 178 1551
vid_4_1820.jpg 179 1551
vid_4_18200.jpg 180 1366
vid_4_18320.jpg 181 1128
vid_4_18360.jpg 182 1275
vid_4_18340.jpg 183 1167
vid_4_1840.jpg 184 1718
vid_4_1840.jpg 185 1718
vid_4_1860.jpg 186 1638
vid_4_1860.jpg 187 1638
vid_4_1860.jpg 188 1638
vid_4_1880.jpg 189 1732
vid_4_1880.jpg 190 1732
vid_4_1880.jpg 191 1732
vid_4_18820.jpg 192 1419
vid_4_18840.jpg 193 1478
vid_4_18860.jpg 194 1592
vid_4_18860.jpg 195 1592
vid_4_18880.jpg 196 1415
vid_4_1900.jpg 197 1470
vid_4_19040.jpg 198 1458
vid_4_19060.jpg 199 1352
vid_4_19080.jpg 200 1460
```

```
vid_4_1920.jpg 201 1526
vid_4_1920.jpg 202 1526
vid_4_1940.jpg 203 1625
vid_4_1940.jpg 204 1625
vid_4_1960.jpg 205 1701
vid_4_1960.jpg 206 1701
vid_4_19760.jpg 207 1439
vid_4_19780.jpg 208 1281
vid_4_19740.jpg 209 1585
vid_4_1980.jpg 210 1692
vid_4_1980.jpg 211 1692
vid_4_1980.jpg 212 1692
vid_4_19880.jpg 213 1442
vid_4_19900.jpg 214 1468
vid_4_19920.jpg 215 1761
vid_4_2000.jpg 216 1943
vid_4_2000.jpg 217 1943
vid_4_2000.jpg 218 1943
vid_4_2000.jpg 219 1943
vid_4_2000.jpg 220 1943
vid_4_2020.jpg 221 1749
vid_4_2020.jpg 222 1749
vid_4_2020.jpg 223 1749
vid_4_2020.jpg 224 1749
vid_4_2040.jpg 225 1869
vid_4_2040.jpg 226 1869
vid_4_2040.jpg 227 1869
vid_4_2040.jpg 228 1869
vid_4_2060.jpg 229 1770
vid_4_2060.jpg 230 1770
vid_4_2060.jpg 231 1770
vid_4_2060.jpg 232 1770
vid_4_2080.jpg 233 1493
vid_4_2080.jpg 234 1493
vid_4_2080.jpg 235 1493
vid_4_2080.jpg 236 1493
vid_4_2100.jpg 237 1504
vid_4_2100.jpg 238 1504
vid_4_2100.jpg 239 1504
vid_4_21160.jpg 240 1194
vid_4_21180.jpg 241 1097
vid_4_2120.jpg 242 1532
vid_4_2120.jpg 243 1532
vid_4_2120.jpg 244 1532
vid_4_21200.jpg 245 1181
vid_4_21220.jpg 246 1355
vid_4_21240.jpg 247 1563
vid_4_21240.jpg 248 1563
```

```
vid_4_21260.jpg 249 1697
vid_4_21260.jpg 250 1697
vid_4_21280.jpg 251 1642
vid_4_21300.jpg 252 1738
vid_4_21320.jpg 253 1639
vid_4_2140.jpg 254 1733
vid_4_2140.jpg 255 1733
vid_4_2140.jpg 256 1733
vid_4_21420.jpg 257 1833
vid_4_21400.jpg 258 1875
vid_4_21440.jpg 259 1823
vid_4_21480.jpg 260 1846
vid_4_21480.jpg 261 1846
vid_4_21500.jpg 262 1794
vid_4_21500.jpg 263 1794
vid_4_21520.jpg 264 1541
vid_4_21560.jpg 265 1752
vid_4_21560.jpg 266 1752
vid_4_21540.jpg 267 1638
vid_4_21580.jpg 268 1644
vid_4_21620.jpg 269 1659
vid_4_21600.jpg 270 1537
vid_4_21640.jpg 271 1752
vid_4_21640.jpg 272 1752
vid_4_2160.jpg 273 1988
vid_4_2160.jpg 274 1988
vid_4_2160.jpg 275 1988
vid_4_21660.jpg 276 1641
vid_4_21680.jpg 277 1747
vid_4_2180.jpg 278 1669
vid_4_2200.jpg 279 1606
vid_4_22240.jpg 280 1595
vid_4_22220.jpg 281 1503
vid_4_22560.jpg 282 1551
vid_4_22540.jpg 283 1415
vid_4_22580.jpg 284 1425
vid_4_22640.jpg 285 1315
vid_4_22660.jpg 286 1377
vid_4_22700.jpg 287 1367
vid_4_22720.jpg 288 1532
vid_4_22740.jpg 289 1389
vid_4_22980.jpg 290 1545
vid_4_23000.jpg 291 1520
vid_4_23020.jpg 292 1461
vid_4_2380.jpg 293 1296
vid_4_2400.jpg 294 1257
vid_4_2420.jpg 295 1309
vid_4_24760.jpg 296 1932
```

```
vid_4_2520.jpg 297 1332
vid_4_2540.jpg 298 1291
vid_4_26300.jpg 299 1251
vid_4_26320.jpg 300 1473
vid_4_26320.jpg 301 1473
vid_4_26340.jpg 302 1691
vid_4_26340.jpg 303 1691
vid_4_26340.jpg 304 1691
vid_4_26360.jpg 305 1714
vid_4_26360.jpg 306 1714
vid_4_26360.jpg 307 1714
vid_4_26360.jpg 308 1714
vid_4_26380.jpg 309 1519
vid_4_26380.jpg 310 1519
vid_4_26380.jpg 311 1519
vid_4_26380.jpg 312 1519
vid_4_26380.jpg 313 1519
vid_4_26420.jpg 314 1365
vid_4_26420.jpg 315 1365
vid_4_26420.jpg 316 1365
vid_4_26420.jpg 317 1365
vid_4_26420.jpg 318 1365
vid_4_26400.jpg 319 1502
vid_4_26400.jpg 320 1502
vid_4_26400.jpg 321 1502
vid_4_26400.jpg 322 1502
vid_4_26400.jpg 323 1502
vid_4_26440.jpg 324 1291
vid_4_26440.jpg 325 1291
vid_4_26440.jpg 326 1291
vid_4_26440.jpg 327 1291
vid_4_26440.jpg 328 1291
vid_4_26480.jpg 329 1255
vid_4_26480.jpg 330 1255
vid_4_26480.jpg 331 1255
vid_4_26480.jpg 332 1255
vid_4_26500.jpg 333 1303
vid_4_26500.jpg 334 1303
vid_4_26500.jpg 335 1303
vid_4_26500.jpg 336 1303
vid_4_26460.jpg 337 1257
vid_4_26460.jpg 338 1257
vid_4_26460.jpg 339 1257
vid_4_26460.jpg 340 1257
vid_4_26460.jpg 341 1257
vid_4_26460.jpg 342 1257
vid_4_26460.jpg 343 1257
vid_4_26520.jpg 344 1268
```

```
vid_4_26520.jpg 345 1268
vid_4_26520.jpg 346 1268
vid_4_26540.jpg 347 1114
vid_4_26540.jpg 348 1114
vid_4_26560.jpg 349 1126
vid_4_26580.jpg 350 1022
vid_4_28220.jpg 351 1508
vid_4_28220.jpg 352 1508
vid_4_28200.jpg 353 1578
vid_4_28200.jpg 354 1578
vid_4_28240.jpg 355 1486
vid_4_28440.jpg 356 1446
vid_4_28840.jpg 357 1680
vid_4_28820.jpg 358 1583
vid_4_28880.jpg 359 1672
vid_4_28860.jpg 360 1724
vid_4_29460.jpg 361 1324
vid_4_29480.jpg 362 1238
vid_4_29500.jpg 363 1270
vid_4_29520.jpg 364 1496
vid_4_29520.jpg 365 1496
vid_4_29540.jpg 366 1375
vid_4_29560.jpg 367 1315
vid_4_29880.jpg 368 1374
vid_4_29880.jpg 369 1374
vid_4_29880.jpg 370 1374
vid_4_29880.jpg 371 1374
vid_4_29900.jpg 372 1349
vid_4_29900.jpg 373 1349
vid_4_29900.jpg 374 1349
vid_4_29920.jpg 375 1414
vid_4_29920.jpg 376 1414
vid_4_29920.jpg 377 1414
vid_4_29920.jpg 378 1414
vid_4_29940.jpg 379 1403
vid_4_29940.jpg 380 1403
vid_4_29940.jpg 381 1403
vid_4_30000.jpg 382 1315
vid_4_30000.jpg 383 1315
vid_4_29980.jpg 384 1323
vid_4_29960.jpg 385 1355
vid_4_29960.jpg 386 1355
vid_4_29960.jpg 387 1355
vid_4_30020.jpg 388 1415
vid_4_30020.jpg 389 1415
vid_4_30440.jpg 390 1497
vid_4_3120.jpg 391 1535
vid_4_3160.jpg 392 1685
```

```
vid_4_3160.jpg 393 1685
vid_4_3140.jpg 394 1630
vid_4_3180.jpg 395 1732
vid_4_3180.jpg 396 1732
vid_4_3200.jpg 397 1653
vid_4_3220.jpg 398 1616
vid_4_3240.jpg 399 1792
vid_4_3260.jpg 400 1663
vid_4_3380.jpg 401 1639
vid_4_3360.jpg 402 1659
vid_4_3340.jpg 403 1675
vid_4_3440.jpg 404 1361
vid_4_3460.jpg 405 1542
vid_4_3520.jpg 406 1475
vid_4_3540.jpg 407 1496
vid_4_3560.jpg 408 1500
vid_4_3820.jpg 409 1320
vid_4_3800.jpg 410 1267
vid_4_3840.jpg 411 1455
vid_4_4540.jpg 412 1761
vid_4_4520.jpg 413 1694
vid_4_4560.jpg 414 1782
vid_4_600.jpg 415 1442
vid_4_6160.jpg 416 1508
vid_4_6160.jpg 417 1508
vid_4_6180.jpg 418 1672
vid_4_6180.jpg 419 1672
vid_4_6200.jpg 420 1814
vid_4_6200.jpg 421 1814
vid_4_620.jpg 422 1426
vid_4_6220.jpg 423 1649
vid_4_6220.jpg 424 1649
vid_4_6220.jpg 425 1649
vid_4_6220.jpg 426 1649
vid_4_6240.jpg 427 1570
vid_4_6240.jpg 428 1570
vid_4_6240.jpg 429 1570
vid_4_6240.jpg 430 1570
vid_4_6240.jpg 431 1570
vid_4_6240.jpg 432 1570
vid_4_6260.jpg 433 1641
vid_4_6260.jpg 434 1641
vid_4_6260.jpg 435 1641
vid_4_6260.jpg 436 1641
vid_4_6260.jpg 437 1641
vid_4_6280.jpg 438 1527
vid_4_6280.jpg 439 1527
vid_4_6280.jpg 440 1527
```

```
vid_4_6280.jpg 441 1527
vid_4_6280.jpg 442 1527
vid_4_6320.jpg 443 1334
vid_4_6320.jpg 444 1334
vid_4_6300.jpg 445 1443
vid_4_6300.jpg 446 1443
vid_4_6300.jpg 447 1443
vid_4_6300.jpg 448 1443
vid_4_6340.jpg 449 1373
vid_4_6340.jpg 450 1373
vid_4_6360.jpg 451 1389
vid_4_6360.jpg 452 1389
vid_4_6380.jpg 453 1353
vid_4_6380.jpg 454 1353
vid_4_6400.jpg 455 1405
vid_4_6400.jpg 456 1405
vid_4_6420.jpg 457 1292
vid_4_6420.jpg 458 1292
vid_4_6420.jpg 459 1292
vid_4_6440.jpg 460 1379
vid_4_6440.jpg 461 1379
vid_4_6440.jpg 462 1379
vid_4_6460.jpg 463 1332
vid_4_6460.jpg 464 1332
vid_4_6480.jpg 465 1275
vid_4_6480.jpg 466 1275
vid_4_6500.jpg 467 1241
vid_4_6520.jpg 468 1240
vid_4_680.jpg 469 1822
vid_4_700.jpg 470 1925
vid_4_700.jpg 471 1925
vid_4_720.jpg 472 1827
vid_4_720.jpg 473 1827
vid_4_740.jpg 474 1672
vid_4_8240.jpg 475 1796
vid_4_8220.jpg 476 1738
vid_4_8260.jpg 477 1770
vid_4_8280.jpg 478 1781
vid_4_8320.jpg 479 1717
vid_4_8340.jpg 480 1615
vid_4_8300.jpg 481 1803
vid_4_8560.jpg 482 1624
vid_4_860.jpg 483 1371
vid_4_8580.jpg 484 1664
vid_4_8600.jpg 485 1697
vid_4_8640.jpg 486 1874
vid_4_8660.jpg 487 1793
vid_4_8680.jpg 488 1656
```

```
vid_4_8700.jpg 489 1718
vid_4_8720.jpg 490 1679
vid_4_8740.jpg 491 1678
vid_4_880.jpg 492 1308
vid_4_8960.jpg 493 1260
vid_4_8980.jpg 494 1301
vid_4_9000.jpg 495 1388
vid_4_900.jpg 496 1235
vid_4_9020.jpg 497 1518
vid_4_9040.jpg 498 1409
vid_4_9040.jpg 499 1409
vid_4_9060.jpg 500 1352
vid_4_9060.jpg 501 1352
vid_4_9080.jpg 502 1301
vid_4_9080.jpg 503 1301
vid_4_920.jpg 504 1387
vid_4_9200.jpg 505 1482
vid_4_9220.jpg 506 1359
vid_4_9240.jpg 507 1478
vid_4_9260.jpg 508 1461
vid_4_9260.jpg 509 1461
vid_4_9280.jpg 510 1384
vid_4_9280.jpg 511 1384
vid_4_9300.jpg 512 1382
vid_4_9300.jpg 513 1382
vid_4_9320.jpg 514 1498
vid_4_9320.jpg 515 1498
vid_4_9340.jpg 516 1362
vid_4_940.jpg 517 1381
vid_4_940.jpg 518 1381
vid_4_9420.jpg 519 1680
vid_4_9460.jpg 520 1565
vid_4_9440.jpg 521 1620
vid_4_9500.jpg 522 1555
vid_4_9520.jpg 523 1545
vid_4_9520.jpg 524 1545
vid_4_9540.jpg 525 1515
vid_4_9540.jpg 526 1515
vid_4_9560.jpg 527 1566
vid_4_9560.jpg 528 1566
vid_4_9560.jpg 529 1566
vid_4_9580.jpg 530 1558
vid_4_9580.jpg 531 1558
vid_4_960.jpg 532 1236
vid_4_9600.jpg 533 1634
vid_4_9600.jpg 534 1634
vid_4_9640.jpg 535 1724
vid_4_9640.jpg 536 1724
```

```
vid_4_9660.jpg 537 1755
vid_4_9620.jpg 538 1752
vid_4_9620.jpg 539 1752
vid_4_9620.jpg 540 1752
vid_4_9620.jpg 541 1752
vid_4_9720.jpg 542 1665
vid_4_9760.jpg 543 1745
vid_4_9760.jpg 544 1745
vid_4_9760.jpg 545 1745
vid_4_9740.jpg 546 1574
vid_4_9700.jpg 547 1676
vid_4_9780.jpg 548 1484
vid_4_9780.jpg 549 1484
vid_4_980.jpg 550 1255
vid_4_9800.jpg 551 1530
vid_4_9800.jpg 552 1530
vid_4_9820.jpg 553 1492
vid_4_9840.jpg 554 1551
vid_4_9860.jpg 555 1590
vid_4_9880.jpg 556 1488
vid_4_9900.jpg 557 1551
vid_4_9960.jpg 558 1384
vid_4_9980.jpg 559 1439
```

[8]:

```
len(image_liste)
```

[8]:

```
10591
```

### 1.3 Convolutional Neural Network

In this section, we will perform two CNN: VGG and LeNet. Then, we will compare the accuracy of them.

[9]:

```
data=[]
data_label=[]
for features,label in image_liste:
    data.append(features)
    data_label.append(label)
```

[10]:

```
data=np.asarray(data)
data_label=np.asarray(data_label)

data.shape
data_label.shape
```

[10]:

```
(10591,)
```

### 1.3.1 Split Test and Train set

[11]:
```
x_train,x_val,y_train,y_val=train_test_split(data, data_label, test_size=0.33,␣
↪random_state=0)
```

### 1.3.2 VGG

[12]:
```
base_model=tf.keras.applications.
↪VGG16(include_top=False,input_shape=(224,224,3),weights='imagenet')
```

```
2022-02-20 14:32:13.942721: I tensorflow/core/platform/cpu_feature_guard.cc:145]
This TensorFlow binary is optimized with Intel(R) MKL-DNN to use the following
CPU instructions in performance critical operations:  SSE4.1 SSE4.2 AVX AVX2 FMA
To enable them in non-MKL-DNN operations, rebuild TensorFlow with the
appropriate compiler flags.
2022-02-20 14:32:13.947318: I
tensorflow/core/common_runtime/process_util.cc:115] Creating new thread pool
with default inter op setting: 8. Tune using inter_op_parallelism_threads for
best performance.
```

[13]:
```
base_model.summary()
```

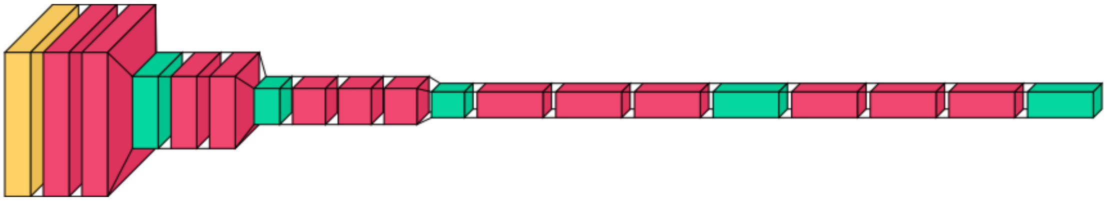
```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
```

```
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
_________________________________________________________________
```

[14]:

```
visualkeras.layered_view(base_model, scale_xy=0.5)
```

[14]:



[15]:

```
model=tf.keras.Sequential()
model.add(base_model)
model.add(tf.keras.layers.GlobalAveragePooling2D())
model.add(tf.keras.layers.Dropout(0.5))
model.add(tf.keras.layers.Dense(1,activation='sigmoid'))
```

[16]:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Model)                (None, 7, 7, 512)         14714688
_________________________________________________________________
global_average_pooling2d (Gl (None, 512)               0
_________________________________________________________________
```

```
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 1)                 513
=================================================================
Total params: 14,715,201
Trainable params: 14,715,201
Non-trainable params: 0
_________________________________________________________________
```

[17]:
```
base_model.trainable=False
```

[18]:
```
for i,layer in enumerate(base_model.layers):
    print(i,layer.name,'-',layer.trainable)
```

```
0 input_1 - False
1 block1_conv1 - False
2 block1_conv2 - False
3 block1_pool - False
4 block2_conv1 - False
5 block2_conv2 - False
6 block2_pool - False
7 block3_conv1 - False
8 block3_conv2 - False
9 block3_conv3 - False
10 block3_pool - False
11 block4_conv1 - False
12 block4_conv2 - False
13 block4_conv3 - False
14 block4_pool - False
15 block5_conv1 - False
16 block5_conv2 - False
17 block5_conv3 - False
18 block5_pool - False
```

[19]:
```
model.compile(loss='binary_crossentropy',optimizer=tf.keras.optimizers.
 ↪Adam(),metrics=['accuracy'])
```

[20]:
```
epoch=3
hist = model.fit(x_train,y_train,epochs=epoch,validation_data=(x_val,y_val))
```

```
Train on 7095 samples, validate on 3496 samples
Epoch 1/3
7095/7095 [==============================] - 3537s 499ms/sample - loss: 0.6029 -
accuracy: 0.8368 - val_loss: 0.1595 - val_accuracy: 0.9531
Epoch 2/3
7095/7095 [==============================] - 3445s 486ms/sample - loss: 0.2652 -
accuracy: 0.9308 - val_loss: 0.1365 - val_accuracy: 0.9688
Epoch 3/3
```

```
7095/7095 [==============================] - 5004s 705ms/sample - loss: 0.2106 -
accuracy: 0.9388 - val_loss: 0.1227 - val_accuracy: 0.9671
```

### 1.3.3 LeNet

Normally, LeNet is used for black-white image or 32x32x1-shape image. But in this task, we extended LeNet to 32x32x3 because the images we got are in rgb scale.

[59]:

```
image_liste=[]
k=0
l=0
z=0 #Loading
for a in pd.read_csv('data/train_solution_bounding.csv').values:
    Name,xmin,ymin,xmax,ymax=a
    bb1={'x1':int(xmin),'y1':int(ymin),'x2':int(xmax),'y2':int(ymax)}
    try:
        img=cv2.imread('data/training_images/'+Name)
        ss.setBaseImage(img)
        ss.switchToSelectiveSearchFast()
        rects = ss.process()
        for i in rects:
            x, y, w, h = i # Selective bounty boxxes
            bb2={'x1':x,'y1':y,'x2':x+w,'y2':y+h}
            img1=img[bb2['y1']:bb2['y2'],bb2['x1']:bb2['x2']] # Crop img
            img1_shape=cv2.resize(img1,(32,32))
            img1_shape = cv2.cvtColor(img1_shape, cv2.COLOR_BGR2GRAY)

            if k<l:
                if 0.5<get_iou(bb1,bb2):
                    image_liste.append([img1_shape,1])
                    k+=1
            else:
                if 0.5<get_iou(bb1,bb2):
                    image_liste.append([img1_shape,1])
                    k+=1
                else:
                    image_liste.append([img1_shape,0])
                    l+=1
    except Exception as e:
        print('Error',e)
    z+=1
    print(Name,z,len(rects))
```

```
vid_4_1000.jpg 1 1133
vid_4_10000.jpg 2 1426
vid_4_10040.jpg 3 1446
vid_4_10020.jpg 4 1402
vid_4_10060.jpg 5 1475
```

```
vid_4_10100.jpg 6 1712
vid_4_10120.jpg 7 1531
vid_4_10140.jpg 8 1630
vid_4_1020.jpg 9 1120
vid_4_1040.jpg 10 1220
vid_4_10480.jpg 11 1714
vid_4_10500.jpg 12 1771
vid_4_10520.jpg 13 1724
vid_4_1060.jpg 14 1323
vid_4_10960.jpg 15 1865
vid_4_10980.jpg 16 1816
vid_4_11000.jpg 17 1770
vid_4_11020.jpg 18 1657
vid_4_11240.jpg 19 1725
vid_4_11260.jpg 20 1912
vid_4_11280.jpg 21 1969
vid_4_11380.jpg 22 1779
vid_4_11400.jpg 23 1851
vid_4_11420.jpg 24 1904
vid_4_11440.jpg 25 1842
vid_4_11900.jpg 26 1973
vid_4_11880.jpg 27 1881
vid_4_11920.jpg 28 1935
vid_4_11940.jpg 29 1917
vid_4_11960.jpg 30 2034
vid_4_11980.jpg 31 2081
vid_4_12000.jpg 32 1848
vid_4_12040.jpg 33 2052
vid_4_12100.jpg 34 1917
vid_4_12060.jpg 35 2009
vid_4_12080.jpg 36 2034
vid_4_12120.jpg 37 1863
vid_4_12140.jpg 38 1665
vid_4_12140.jpg 39 1665
vid_4_12160.jpg 40 1697
vid_4_12160.jpg 41 1697
vid_4_12160.jpg 42 1697
vid_4_12180.jpg 43 1599
vid_4_12180.jpg 44 1599
vid_4_12180.jpg 45 1599
vid_4_12200.jpg 46 1399
vid_4_12200.jpg 47 1399
vid_4_12220.jpg 48 1479
vid_4_12240.jpg 49 1483
vid_4_12240.jpg 50 1483
vid_4_12260.jpg 51 1963
vid_4_12260.jpg 52 1963
vid_4_12280.jpg 53 2099
```

```
vid_4_12280.jpg 54 2099
vid_4_12300.jpg 55 2079
vid_4_12300.jpg 56 2079
vid_4_12300.jpg 57 2079
vid_4_12300.jpg 58 2079
vid_4_12320.jpg 59 1961
vid_4_12320.jpg 60 1961
vid_4_12320.jpg 61 1961
vid_4_12340.jpg 62 1904
vid_4_12340.jpg 63 1904
vid_4_12340.jpg 64 1904
vid_4_12360.jpg 65 1914
vid_4_12360.jpg 66 1914
vid_4_12380.jpg 67 1945
vid_4_12480.jpg 68 2043
vid_4_13580.jpg 69 1833
vid_4_13620.jpg 70 1981
vid_4_13640.jpg 71 2000
vid_4_13640.jpg 72 2000
vid_4_13660.jpg 73 2193
vid_4_13660.jpg 74 2193
vid_4_13680.jpg 75 2192
vid_4_13680.jpg 76 2192
vid_4_13680.jpg 77 2192
vid_4_13700.jpg 78 2305
vid_4_13700.jpg 79 2305
vid_4_13700.jpg 80 2305
vid_4_13700.jpg 81 2305
vid_4_13720.jpg 82 2107
vid_4_13720.jpg 83 2107
vid_4_13720.jpg 84 2107
vid_4_13740.jpg 85 2135
vid_4_13740.jpg 86 2135
vid_4_13760.jpg 87 2246
vid_4_13760.jpg 88 2246
vid_4_13760.jpg 89 2246
vid_4_13780.jpg 90 2297
vid_4_13780.jpg 91 2297
vid_4_13800.jpg 92 2199
vid_4_13800.jpg 93 2199
vid_4_13840.jpg 94 2303
vid_4_13840.jpg 95 2303
vid_4_13820.jpg 96 2265
vid_4_13820.jpg 97 2265
vid_4_13820.jpg 98 2265
vid_4_13860.jpg 99 2392
vid_4_13860.jpg 100 2392
vid_4_13860.jpg 101 2392
```

```
vid_4_13900.jpg 102 2250
vid_4_13880.jpg 103 2451
vid_4_13880.jpg 104 2451
vid_4_13920.jpg 105 2416
vid_4_14140.jpg 106 2160
vid_4_14160.jpg 107 1787
vid_4_14360.jpg 108 1561
vid_4_14340.jpg 109 1403
vid_4_14380.jpg 110 1532
vid_4_14400.jpg 111 1634
vid_4_14460.jpg 112 1247
vid_4_14440.jpg 113 1455
vid_4_14480.jpg 114 1173
vid_4_14500.jpg 115 1333
vid_4_15000.jpg 116 1832
vid_4_15020.jpg 117 1736
vid_4_15040.jpg 118 1843
vid_4_16000.jpg 119 1406
vid_4_16020.jpg 120 1369
vid_4_16040.jpg 121 1428
vid_4_16060.jpg 122 1359
vid_4_16100.jpg 123 1293
vid_4_16120.jpg 124 1410
vid_4_16140.jpg 125 1450
vid_4_16160.jpg 126 1383
vid_4_16180.jpg 127 1284
vid_4_16280.jpg 128 1450
vid_4_16320.jpg 129 1341
vid_4_16300.jpg 130 1517
vid_4_16400.jpg 131 1148
vid_4_16420.jpg 132 1196
vid_4_16500.jpg 133 1358
vid_4_16660.jpg 134 1526
vid_4_16680.jpg 135 1480
vid_4_16700.jpg 136 1619
vid_4_16720.jpg 137 1449
vid_4_17040.jpg 138 1571
vid_4_17060.jpg 139 1536
vid_4_17080.jpg 140 1581
vid_4_17100.jpg 141 1504
vid_4_17140.jpg 142 1486
vid_4_17140.jpg 143 1486
vid_4_17120.jpg 144 1433
vid_4_17160.jpg 145 1515
vid_4_17180.jpg 146 1513
vid_4_17240.jpg 147 1408
vid_4_17260.jpg 148 1249
vid_4_17320.jpg 149 1365
```

```
vid_4_17340.jpg 150 1474
vid_4_17340.jpg 151 1474
vid_4_17360.jpg 152 1370
vid_4_17400.jpg 153 1485
vid_4_17440.jpg 154 1501
vid_4_17420.jpg 155 1443
vid_4_17560.jpg 156 1679
vid_4_17560.jpg 157 1679
vid_4_17560.jpg 158 1679
vid_4_17540.jpg 159 1707
vid_4_17540.jpg 160 1707
vid_4_17580.jpg 161 1604
vid_4_17580.jpg 162 1604
vid_4_1760.jpg 163 1370
vid_4_17600.jpg 164 1789
vid_4_17620.jpg 165 1811
vid_4_17620.jpg 166 1811
vid_4_17640.jpg 167 1883
vid_4_17640.jpg 168 1883
vid_4_17660.jpg 169 1794
vid_4_17660.jpg 170 1794
vid_4_17680.jpg 171 1649
vid_4_1780.jpg 172 1500
vid_4_1780.jpg 173 1500
vid_4_1800.jpg 174 1613
vid_4_1800.jpg 175 1613
vid_4_1800.jpg 176 1613
vid_4_18180.jpg 177 1378
vid_4_1820.jpg 178 1551
vid_4_1820.jpg 179 1551
vid_4_18200.jpg 180 1366
vid_4_18320.jpg 181 1128
vid_4_18360.jpg 182 1275
vid_4_18340.jpg 183 1167
vid_4_1840.jpg 184 1718
vid_4_1840.jpg 185 1718
vid_4_1860.jpg 186 1638
vid_4_1860.jpg 187 1638
vid_4_1860.jpg 188 1638
vid_4_1880.jpg 189 1732
vid_4_1880.jpg 190 1732
vid_4_1880.jpg 191 1732
vid_4_18820.jpg 192 1419
vid_4_18840.jpg 193 1478
vid_4_18860.jpg 194 1592
vid_4_18860.jpg 195 1592
vid_4_18880.jpg 196 1415
vid_4_1900.jpg 197 1470
```

```
vid_4_19040.jpg 198 1458
vid_4_19060.jpg 199 1352
vid_4_19080.jpg 200 1460
vid_4_1920.jpg 201 1526
vid_4_1920.jpg 202 1526
vid_4_1940.jpg 203 1625
vid_4_1940.jpg 204 1625
vid_4_1960.jpg 205 1701
vid_4_1960.jpg 206 1701
vid_4_19760.jpg 207 1439
vid_4_19780.jpg 208 1281
vid_4_19740.jpg 209 1585
vid_4_1980.jpg 210 1692
vid_4_1980.jpg 211 1692
vid_4_1980.jpg 212 1692
vid_4_19880.jpg 213 1442
vid_4_19900.jpg 214 1468
vid_4_19920.jpg 215 1761
vid_4_2000.jpg 216 1943
vid_4_2000.jpg 217 1943
vid_4_2000.jpg 218 1943
vid_4_2000.jpg 219 1943
vid_4_2000.jpg 220 1943
vid_4_2020.jpg 221 1749
vid_4_2020.jpg 222 1749
vid_4_2020.jpg 223 1749
vid_4_2020.jpg 224 1749
vid_4_2040.jpg 225 1869
vid_4_2040.jpg 226 1869
vid_4_2040.jpg 227 1869
vid_4_2040.jpg 228 1869
vid_4_2060.jpg 229 1770
vid_4_2060.jpg 230 1770
vid_4_2060.jpg 231 1770
vid_4_2060.jpg 232 1770
vid_4_2080.jpg 233 1493
vid_4_2080.jpg 234 1493
vid_4_2080.jpg 235 1493
vid_4_2080.jpg 236 1493
vid_4_2100.jpg 237 1504
vid_4_2100.jpg 238 1504
vid_4_2100.jpg 239 1504
vid_4_21160.jpg 240 1194
vid_4_21180.jpg 241 1097
vid_4_2120.jpg 242 1532
vid_4_2120.jpg 243 1532
vid_4_2120.jpg 244 1532
vid_4_21200.jpg 245 1181
```

```
vid_4_21220.jpg 246 1355
vid_4_21240.jpg 247 1563
vid_4_21240.jpg 248 1563
vid_4_21260.jpg 249 1697
vid_4_21260.jpg 250 1697
vid_4_21280.jpg 251 1642
vid_4_21300.jpg 252 1738
vid_4_21320.jpg 253 1639
vid_4_2140.jpg 254 1733
vid_4_2140.jpg 255 1733
vid_4_2140.jpg 256 1733
vid_4_21420.jpg 257 1833
vid_4_21400.jpg 258 1875
vid_4_21440.jpg 259 1823
vid_4_21480.jpg 260 1846
vid_4_21480.jpg 261 1846
vid_4_21500.jpg 262 1794
vid_4_21500.jpg 263 1794
vid_4_21520.jpg 264 1541
vid_4_21560.jpg 265 1752
vid_4_21560.jpg 266 1752
vid_4_21540.jpg 267 1638
vid_4_21580.jpg 268 1644
vid_4_21620.jpg 269 1659
vid_4_21600.jpg 270 1537
vid_4_21640.jpg 271 1752
vid_4_21640.jpg 272 1752
vid_4_2160.jpg 273 1988
vid_4_2160.jpg 274 1988
vid_4_2160.jpg 275 1988
vid_4_21660.jpg 276 1641
vid_4_21680.jpg 277 1747
vid_4_2180.jpg 278 1669
vid_4_2200.jpg 279 1606
vid_4_22240.jpg 280 1595
vid_4_22220.jpg 281 1503
vid_4_22560.jpg 282 1551
vid_4_22540.jpg 283 1415
vid_4_22580.jpg 284 1425
vid_4_22640.jpg 285 1315
vid_4_22660.jpg 286 1377
vid_4_22700.jpg 287 1367
vid_4_22720.jpg 288 1532
vid_4_22740.jpg 289 1389
vid_4_22980.jpg 290 1545
vid_4_23000.jpg 291 1520
vid_4_23020.jpg 292 1461
vid_4_2380.jpg 293 1296
```

```
vid_4_2400.jpg 294 1257
vid_4_2420.jpg 295 1309
vid_4_24760.jpg 296 1932
vid_4_2520.jpg 297 1332
vid_4_2540.jpg 298 1291
vid_4_26300.jpg 299 1251
vid_4_26320.jpg 300 1473
vid_4_26320.jpg 301 1473
vid_4_26340.jpg 302 1691
vid_4_26340.jpg 303 1691
vid_4_26340.jpg 304 1691
vid_4_26360.jpg 305 1714
vid_4_26360.jpg 306 1714
vid_4_26360.jpg 307 1714
vid_4_26360.jpg 308 1714
vid_4_26380.jpg 309 1519
vid_4_26380.jpg 310 1519
vid_4_26380.jpg 311 1519
vid_4_26380.jpg 312 1519
vid_4_26380.jpg 313 1519
vid_4_26420.jpg 314 1365
vid_4_26420.jpg 315 1365
vid_4_26420.jpg 316 1365
vid_4_26420.jpg 317 1365
vid_4_26420.jpg 318 1365
vid_4_26400.jpg 319 1502
vid_4_26400.jpg 320 1502
vid_4_26400.jpg 321 1502
vid_4_26400.jpg 322 1502
vid_4_26400.jpg 323 1502
vid_4_26440.jpg 324 1291
vid_4_26440.jpg 325 1291
vid_4_26440.jpg 326 1291
vid_4_26440.jpg 327 1291
vid_4_26440.jpg 328 1291
vid_4_26480.jpg 329 1255
vid_4_26480.jpg 330 1255
vid_4_26480.jpg 331 1255
vid_4_26480.jpg 332 1255
vid_4_26500.jpg 333 1303
vid_4_26500.jpg 334 1303
vid_4_26500.jpg 335 1303
vid_4_26500.jpg 336 1303
vid_4_26460.jpg 337 1257
vid_4_26460.jpg 338 1257
vid_4_26460.jpg 339 1257
vid_4_26460.jpg 340 1257
vid_4_26460.jpg 341 1257
```

```
vid_4_26460.jpg 342 1257
vid_4_26460.jpg 343 1257
vid_4_26520.jpg 344 1268
vid_4_26520.jpg 345 1268
vid_4_26520.jpg 346 1268
vid_4_26540.jpg 347 1114
vid_4_26540.jpg 348 1114
vid_4_26560.jpg 349 1126
vid_4_26580.jpg 350 1022
vid_4_28220.jpg 351 1508
vid_4_28220.jpg 352 1508
vid_4_28200.jpg 353 1578
vid_4_28200.jpg 354 1578
vid_4_28240.jpg 355 1486
vid_4_28440.jpg 356 1446
vid_4_28840.jpg 357 1680
vid_4_28820.jpg 358 1583
vid_4_28880.jpg 359 1672
vid_4_28860.jpg 360 1724
vid_4_29460.jpg 361 1324
vid_4_29480.jpg 362 1238
vid_4_29500.jpg 363 1270
vid_4_29520.jpg 364 1496
vid_4_29520.jpg 365 1496
vid_4_29540.jpg 366 1375
vid_4_29560.jpg 367 1315
vid_4_29880.jpg 368 1374
vid_4_29880.jpg 369 1374
vid_4_29880.jpg 370 1374
vid_4_29880.jpg 371 1374
vid_4_29900.jpg 372 1349
vid_4_29900.jpg 373 1349
vid_4_29900.jpg 374 1349
vid_4_29920.jpg 375 1414
vid_4_29920.jpg 376 1414
vid_4_29920.jpg 377 1414
vid_4_29920.jpg 378 1414
vid_4_29940.jpg 379 1403
vid_4_29940.jpg 380 1403
vid_4_29940.jpg 381 1403
vid_4_30000.jpg 382 1315
vid_4_30000.jpg 383 1315
vid_4_29980.jpg 384 1323
vid_4_29960.jpg 385 1355
vid_4_29960.jpg 386 1355
vid_4_29960.jpg 387 1355
vid_4_30020.jpg 388 1415
vid_4_30020.jpg 389 1415
```

```
vid_4_30440.jpg 390 1497
vid_4_3120.jpg 391 1535
vid_4_3160.jpg 392 1685
vid_4_3160.jpg 393 1685
vid_4_3140.jpg 394 1630
vid_4_3180.jpg 395 1732
vid_4_3180.jpg 396 1732
vid_4_3200.jpg 397 1653
vid_4_3220.jpg 398 1616
vid_4_3240.jpg 399 1792
vid_4_3260.jpg 400 1663
vid_4_3380.jpg 401 1639
vid_4_3360.jpg 402 1659
vid_4_3340.jpg 403 1675
vid_4_3440.jpg 404 1361
vid_4_3460.jpg 405 1542
vid_4_3520.jpg 406 1475
vid_4_3540.jpg 407 1496
vid_4_3560.jpg 408 1500
vid_4_3820.jpg 409 1320
vid_4_3800.jpg 410 1267
vid_4_3840.jpg 411 1455
vid_4_4540.jpg 412 1761
vid_4_4520.jpg 413 1694
vid_4_4560.jpg 414 1782
vid_4_600.jpg 415 1442
vid_4_6160.jpg 416 1508
vid_4_6160.jpg 417 1508
vid_4_6180.jpg 418 1672
vid_4_6180.jpg 419 1672
vid_4_6200.jpg 420 1814
vid_4_6200.jpg 421 1814
vid_4_620.jpg 422 1426
vid_4_6220.jpg 423 1649
vid_4_6220.jpg 424 1649
vid_4_6220.jpg 425 1649
vid_4_6220.jpg 426 1649
vid_4_6240.jpg 427 1570
vid_4_6240.jpg 428 1570
vid_4_6240.jpg 429 1570
vid_4_6240.jpg 430 1570
vid_4_6240.jpg 431 1570
vid_4_6240.jpg 432 1570
vid_4_6260.jpg 433 1641
vid_4_6260.jpg 434 1641
vid_4_6260.jpg 435 1641
vid_4_6260.jpg 436 1641
vid_4_6260.jpg 437 1641
```

```
vid_4_6280.jpg 438 1527
vid_4_6280.jpg 439 1527
vid_4_6280.jpg 440 1527
vid_4_6280.jpg 441 1527
vid_4_6280.jpg 442 1527
vid_4_6320.jpg 443 1334
vid_4_6320.jpg 444 1334
vid_4_6300.jpg 445 1443
vid_4_6300.jpg 446 1443
vid_4_6300.jpg 447 1443
vid_4_6300.jpg 448 1443
vid_4_6340.jpg 449 1373
vid_4_6340.jpg 450 1373
vid_4_6360.jpg 451 1389
vid_4_6360.jpg 452 1389
vid_4_6380.jpg 453 1353
vid_4_6380.jpg 454 1353
vid_4_6400.jpg 455 1405
vid_4_6400.jpg 456 1405
vid_4_6420.jpg 457 1292
vid_4_6420.jpg 458 1292
vid_4_6420.jpg 459 1292
vid_4_6440.jpg 460 1379
vid_4_6440.jpg 461 1379
vid_4_6440.jpg 462 1379
vid_4_6460.jpg 463 1332
vid_4_6460.jpg 464 1332
vid_4_6480.jpg 465 1275
vid_4_6480.jpg 466 1275
vid_4_6500.jpg 467 1241
vid_4_6520.jpg 468 1240
vid_4_680.jpg 469 1822
vid_4_700.jpg 470 1925
vid_4_700.jpg 471 1925
vid_4_720.jpg 472 1827
vid_4_720.jpg 473 1827
vid_4_740.jpg 474 1672
vid_4_8240.jpg 475 1796
vid_4_8220.jpg 476 1738
vid_4_8260.jpg 477 1770
vid_4_8280.jpg 478 1781
vid_4_8320.jpg 479 1717
vid_4_8340.jpg 480 1615
vid_4_8300.jpg 481 1803
vid_4_8560.jpg 482 1624
vid_4_860.jpg 483 1371
vid_4_8580.jpg 484 1664
vid_4_8600.jpg 485 1697
```

```
vid_4_8640.jpg 486 1874
vid_4_8660.jpg 487 1793
vid_4_8680.jpg 488 1656
vid_4_8700.jpg 489 1718
vid_4_8720.jpg 490 1679
vid_4_8740.jpg 491 1678
vid_4_880.jpg 492 1308
vid_4_8960.jpg 493 1260
vid_4_8980.jpg 494 1301
vid_4_9000.jpg 495 1388
vid_4_900.jpg 496 1235
vid_4_9020.jpg 497 1518
vid_4_9040.jpg 498 1409
vid_4_9040.jpg 499 1409
vid_4_9060.jpg 500 1352
vid_4_9060.jpg 501 1352
vid_4_9080.jpg 502 1301
vid_4_9080.jpg 503 1301
vid_4_920.jpg 504 1387
vid_4_9200.jpg 505 1482
vid_4_9220.jpg 506 1359
vid_4_9240.jpg 507 1478
vid_4_9260.jpg 508 1461
vid_4_9260.jpg 509 1461
vid_4_9280.jpg 510 1384
vid_4_9280.jpg 511 1384
vid_4_9300.jpg 512 1382
vid_4_9300.jpg 513 1382
vid_4_9320.jpg 514 1498
vid_4_9320.jpg 515 1498
vid_4_9340.jpg 516 1362
vid_4_940.jpg 517 1381
vid_4_940.jpg 518 1381
vid_4_9420.jpg 519 1680
vid_4_9460.jpg 520 1565
vid_4_9440.jpg 521 1620
vid_4_9500.jpg 522 1555
vid_4_9520.jpg 523 1545
vid_4_9520.jpg 524 1545
vid_4_9540.jpg 525 1515
vid_4_9540.jpg 526 1515
vid_4_9560.jpg 527 1566
vid_4_9560.jpg 528 1566
vid_4_9560.jpg 529 1566
vid_4_9580.jpg 530 1558
vid_4_9580.jpg 531 1558
vid_4_960.jpg 532 1236
vid_4_9600.jpg 533 1634
```

```
vid_4_9600.jpg 534 1634
vid_4_9640.jpg 535 1724
vid_4_9640.jpg 536 1724
vid_4_9660.jpg 537 1755
vid_4_9620.jpg 538 1752
vid_4_9620.jpg 539 1752
vid_4_9620.jpg 540 1752
vid_4_9620.jpg 541 1752
vid_4_9720.jpg 542 1665
vid_4_9760.jpg 543 1745
vid_4_9760.jpg 544 1745
vid_4_9760.jpg 545 1745
vid_4_9740.jpg 546 1574
vid_4_9700.jpg 547 1676
vid_4_9780.jpg 548 1484
vid_4_9780.jpg 549 1484
vid_4_980.jpg 550 1255
vid_4_9800.jpg 551 1530
vid_4_9800.jpg 552 1530
vid_4_9820.jpg 553 1492
vid_4_9840.jpg 554 1551
vid_4_9860.jpg 555 1590
vid_4_9880.jpg 556 1488
vid_4_9900.jpg 557 1551
vid_4_9960.jpg 558 1384
vid_4_9980.jpg 559 1439
```

[60]:

```
lenet_model = Sequential()
lenet_model.add(Conv2D(6, kernel_size=(5, 5), strides=(1, 1),␣
 ↪activation='relu', input_shape=(32,32,1), padding="same"))
lenet_model.add(AveragePooling2D(pool_size=(2, 2), strides=(2, 2),␣
 ↪padding='valid'))
lenet_model.add(Conv2D(16, kernel_size=(5, 5), strides=(1, 1),␣
 ↪activation='relu', padding='valid'))
lenet_model.add(AveragePooling2D(pool_size=(2, 2), strides=(2, 2),␣
 ↪padding='valid'))
lenet_model.add(Flatten())
lenet_model.add(Dense(120, activation='relu'))
lenet_model.add(Dense(84, activation='relu'))
lenet_model.add(Dense(1, activation='softmax'))

lenet_model.
 ↪compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])

lenet_model.summary()
```

```
Model: "sequential_4"
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_7 (Conv2D)            (None, 32, 32, 6)         156
_________________________________________________________________
average_pooling2d_7 (Average (None, 16, 16, 6)         0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 12, 12, 16)        2416
_________________________________________________________________
average_pooling2d_8 (Average (None, 6, 6, 16)          0
_________________________________________________________________
flatten_4 (Flatten)          (None, 576)               0
_________________________________________________________________
dense_10 (Dense)             (None, 120)               69240
_________________________________________________________________
dense_11 (Dense)             (None, 84)                10164
_________________________________________________________________
dense_12 (Dense)             (None, 1)                 85
=================================================================
Total params: 82,061
Trainable params: 82,061
Non-trainable params: 0
_________________________________________________________________
```

[67]:

```
data=[]
data_label=[]
for features,label in image_liste:
    data.append(features)
    data_label.append(label)

data=np.asarray(data)
data_label=np.asarray(data_label)

data = data.reshape([len(image_liste), 32, 32, 1])
```

[68]:

```
x_train,x_val,y_train,y_val=train_test_split(data, data_label, test_size=0.33,␣
 ↪random_state=0)
```

[69]:

```
epoch=3
hist = lenet_model.
 ↪fit(x_train,y_train,epochs=epoch,validation_data=(x_val,y_val))
```

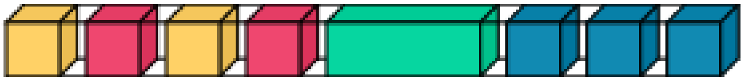
```
Train on 7095 samples, validate on 3496 samples
Epoch 1/3
7095/7095 [==============================] - 7s 1ms/step - loss: 7.7001 -
accuracy: 0.4978 - val_loss: 7.6008 - val_accuracy: 0.5043
Epoch 2/3
7095/7095 [==============================] - 7s 1ms/step - loss: 7.7001 -
```

```
accuracy: 0.4978 - val_loss: 7.6008 - val_accuracy: 0.5043
Epoch 3/3
7095/7095 [==============================] - 7s 998us/step - loss: 7.7001 -
accuracy: 0.4978 - val_loss: 7.6008 - val_accuracy: 0.5043
```

[70]:

```
visualkeras.layered_view(lenet_model, scale_xy=0.5)
```

[70]:



### 1.3.4 Test of VGG

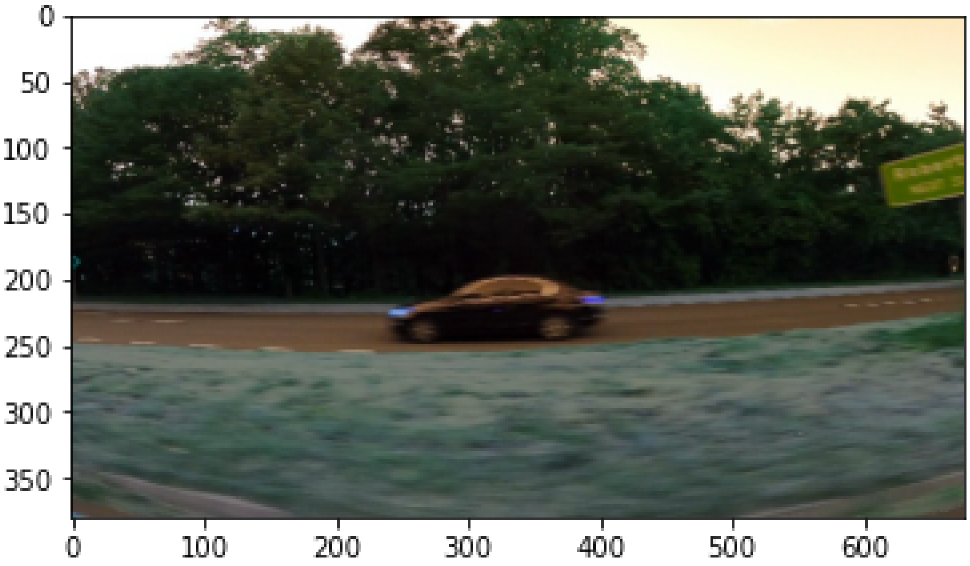
[38]:

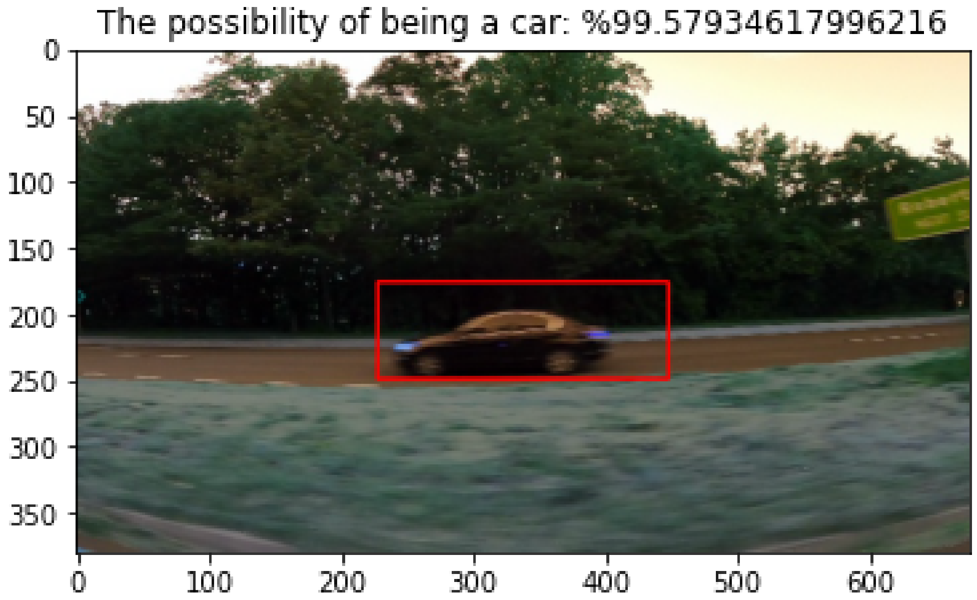
```
car=[]
image = 'data/testing_images/vid_5_26620.jpg'

predict_image = cv2.imread(image)
ss.setBaseImage(predict_image)
ss.switchToSelectiveSearchFast()
segments = ss.process()

print('Total Segments: ',len(segments))
for i in segments:
    x, y, w, h = i
    bb3 = {'x1':x, 'y1':y, 'x2':x+w, 'y2':y+h}
    try:
        assert bb3['x1'] < bb3['x2']
        assert bb3['y1'] < bb3['y2']
        image_data = predict_image[bb3['y1']:bb3['y2'],bb3['x1']:bb3['x2']]
        image_data = cv2.resize(image_data,(224,224))
        image_data = image_data.astype(np.float32)
        prediction_segment = model.predict(image_data.reshape(1,224,224,3))
        if prediction_segment[0]>0.5:
            car.append([bb3,prediction_segment[0]])
        else:
            pass

    except Exception as e:
        print('Error',e)
        print('Possible Cars: ',len(car))
       ␣
 ↪print('-------------------------------------------------------------------------')

predict_image = cv2.imread(image)
car[np.argmax(np.array(car)[:,1])][0]
```

```
pt1 = (car[np.argmax(np.array(car)[:,1])][0]['x1'],car[np.argmax(np.array(car)[:
 ↪,1])][0]['y1'])
pt2 = (car[np.argmax(np.array(car)[:,1])][0]['x2'],car[np.argmax(np.array(car)[:
 ↪,1])][0]['y2'])
plt.figure()
plt.imshow(predict_image)
cv2.rectangle(predict_image,pt1,pt2,(255, 0, 0),2)
plt.figure()
plt.title(f'The possibility of being a car: %{car[np.argmax(np.array(car)[:
 ↪,1])][1][0]*100}')
plt.imshow(predict_image)
```

```
Total Segments:  1515
```

```
[38]: <matplotlib.image.AxesImage at 0x7f8dde6f7a10>
```





### 1.3.5 Test of LeNet

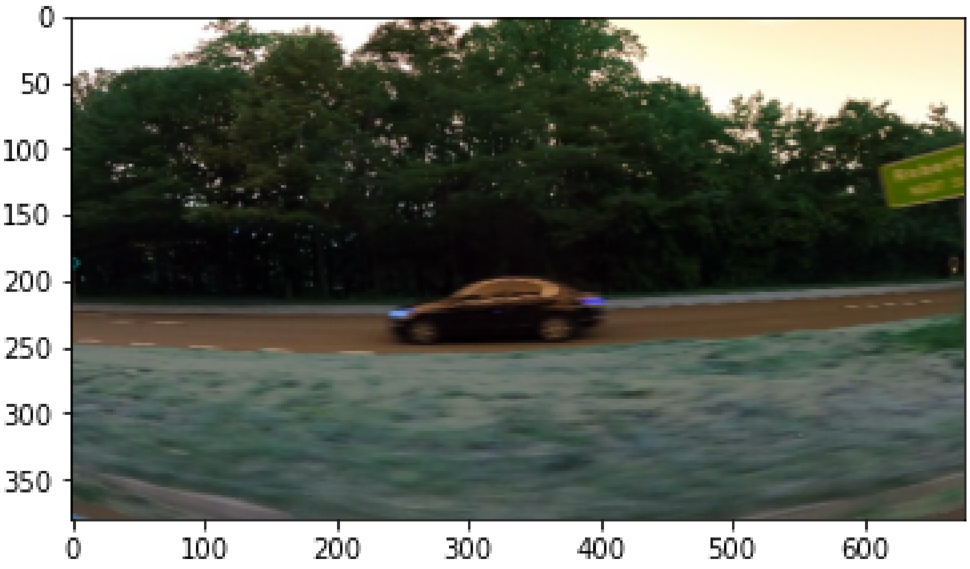
[66]:

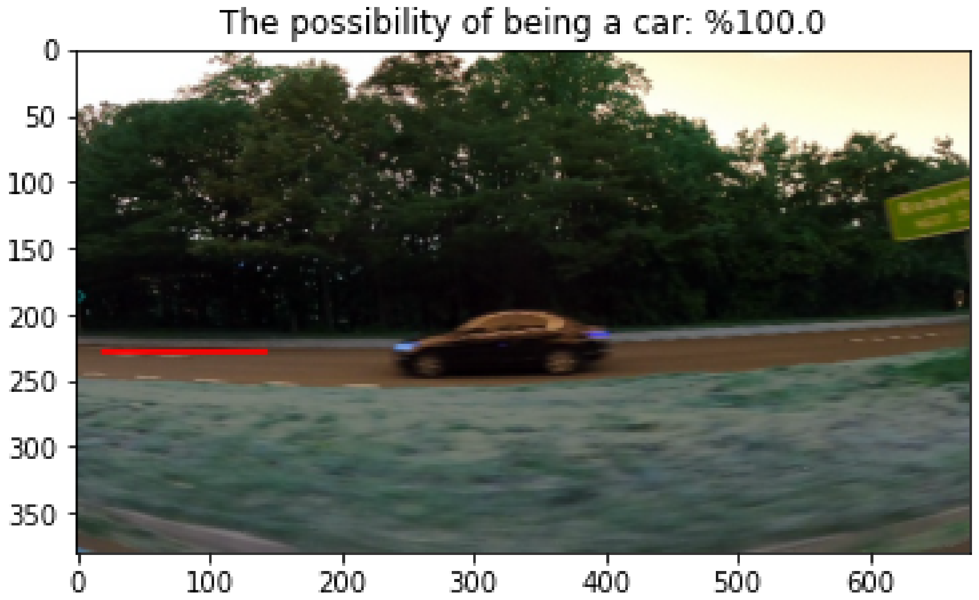
```
car=[]
image = 'data/testing_images/vid_5_26620.jpg'

predict_image = cv2.imread(image)
ss.setBaseImage(predict_image)
ss.switchToSelectiveSearchFast()
segments = ss.process()

print('Total Segments: ',len(segments))
for i in segments:
    x, y, w, h = i
    bb3 = {'x1':x, 'y1':y, 'x2':x+w, 'y2':y+h}
    try:
        assert bb3['x1'] < bb3['x2']
        assert bb3['y1'] < bb3['y2']
        image_data = predict_image[bb3['y1']:bb3['y2'],bb3['x1']:bb3['x2']]
        image_data = cv2.resize(image_data,(32,32))
        image_data = cv2.cvtColor(image_data, cv2.COLOR_BGR2GRAY)
        image_data = image_data.astype(np.float32)
        prediction_segment = lenet_model.predict(image_data.reshape(1,32,32,1))
        if prediction_segment[0]>0.5:
            car.append([bb3,prediction_segment[0]])
        else:
            pass
    except Exception as e:
        print('Error',e)
        print('Possible Cars: ',len(car))
       ␣
 ↪print('-------------------------------------------------------------------------')

predict_image = cv2.imread(image)
car[np.argmax(np.array(car)[:,1])][0]
pt1 = (car[np.argmax(np.array(car)[:,1])][0]['x1'],car[np.argmax(np.array(car)[:
 ↪,1])][0]['y1'])
pt2 = (car[np.argmax(np.array(car)[:,1])][0]['x2'],car[np.argmax(np.array(car)[:
 ↪,1])][0]['y2'])
plt.figure()
plt.imshow(predict_image)
cv2.rectangle(predict_image,pt1,pt2,(255, 0, 0),2)
plt.figure()
plt.title(f'The possibility of being a car: %{car[np.argmax(np.array(car)[:
 ↪,1])][1][0]*100}')
plt.imshow(predict_image)
```

```
Total Segments:  1515
```

[66]:

```
<matplotlib.image.AxesImage at 0x7f8bac088650>
```





#### 1.3.6 Summary and Future Plan

The VGG performed far better than LeNet on car detection of the given dataset. The accuracy of VGG is at 96 percent while the accuracy of LeNet is at 50 percent. The limitation of my laptop is that running VGG with 224x224 images took around an hour for one epoch. The future development could be trying to play with another model with the increase in speed and accuracy.

### 1.4 References

Special Thanks to

1. Step-by-step RCNN
2. Some codes I obtained from
3. LeNet network
4. Neural Network Architecture
5. Intersection Over Union