Computer Vision **(Due: 18/05/2020)**

Implementing advanced Computer Vision Algorithms (Quiz_C3)

Instructor: Tamal Maharaj *Name:* Jimut Bahan Pal

#TASK:

Apply any three Deep Learning algorithms on an image for image classification, detection, and/or segmentation. Write a report with your output and methods used. Discuss the result you get.

Example of methods: LeNet-5, VGG16, VGG 19, Inception, ResNet, SSD, YOLO, Faster R-CNN, U-Net, and so many others. Refer L23 if you need help.

Here, we implemented some of the state of the art Computer Vision algorithms that uses deep learning for performing very hard task mainly related to segmentation, detection and clasification. Since we donot have the computational resources to train the model from scratch, we used weights that are provided by the openCV framework for different kind of models. Here are our results,

DNN basic

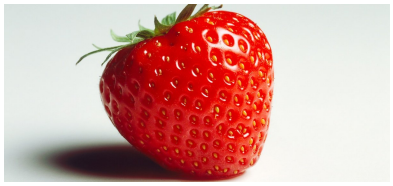
For the purpose of object recognition we used *bvlc_googlenet* from BAIR (Berkeley Artificial Intelligence Research Division), which is a deployable Caffe model. We have run the script **deep_learning_basic.py**, and tested on some of the images. A bluring will be a great way to reduce noise, to the input images [1],

class: triceratops
probability: **0.5667343**
class: curly-coated retriever
probability: 0.07776283
class: starfish, sea star
probability: 0.038735963
class: miniature poodle
probability: 0.029573934
class: Irish water spaniel
probability: 0.024101915

class: Persian cat
probability: **0.15481843**
class: Egyptian cat
probability: 0.08799346
class: tabby, tabby cat
probability: 0.08116494
class: hamster
probability: 0.061302546
class: paper towel
probability: 0.052314423

class: jackfruit, jak, jack
probability: **0.9991935**
class: bolete
probability: 0.0005104809
class: honeycomb
probability: 0.00018546778
class: mushroom
probability: 1.257665e-05
class: hen-of-the-woods, hen of the woods, Polyporus frondosus, Grifola frondosa
probability: 1.1936828e-05

class: croquet ball
probability: **0.47241527**
class: pomegranate
probability: 0.14049406
class: buckeye, horse chestnut, conker
probability: 0.053202752
class: fig
probability: 0.039501205
class: maraca
probability: 0.038151063



class: strawberry
probability: **0.999905**
class: strainer
probability: 1.9462803e-05
class: golf ball
probability: 1.1886338e-05
class: thimble
probability: 1.0093223e-05
class: pomegranate
probability: 6.130343e-06

class: strawberry
probability: **0.9999021**
class: pineapple, ananas
probability: 1.195709e-05
class: daisy
probability: 9.290467e-06
class: banana
probability: 7.900727e-06
class: tray
probability: 7.2159214e-06

class: traffic light, traffic signal, stoplight
probability: **1.0**
class: pole
probability: 7.782289e-09
class: street sign
probability: 7.994616e-10
class: abacus
probability: 1.7547169e-11
class: cinema, movie theater, movie theatre, movie house, picture palace
probability: 1.21549845e-11

From these examples, we conclude that it wrongly classifies mango as croquet ball, which is even complicated for me, since I thought it was some kind of tomato at first glance. It is rare to see riped red mango nowadays.

Mask-RCNN

We have used Mask-RCNN to get pixel wise segmentation. We used the COCO model, more specifically **mask_rcnn_inception_v2_coco_2018_01_28**. The algorithm finds the regions of interest, creates a mask, applies a mask, and creates a bounding box on the images. The result of applying mask R-CNN to an image of 2 cars is shown in Figure 1, when the same is applied to classical image of horse, car, dog, and person, we get the masked image as shown in Figure 2. For the image of Ardian, we get the mask RCNN as shown in Figure 3.

Object Detection using Deep Learning

We have used the MobileNetSSD model for the **deep_learning_object_detection.py** script, which detects the classes which a object belongs to when passed through the algorithm. The images that are detected correctly by the algorithm is shown in Figure 5, 6, 7, 8, 9, and Figure 10, respectively.

YOLO - You Only Look Once

We have used the classic picture, and got all the bounding boxes detected correctly as shown in Figure 4.

References

[1] Jimut Bahan Pal. A deeper look into hybrid images, 2020. https://arxiv.org/abs/2001.11302 available on web, last accessed on May 20, 2020.

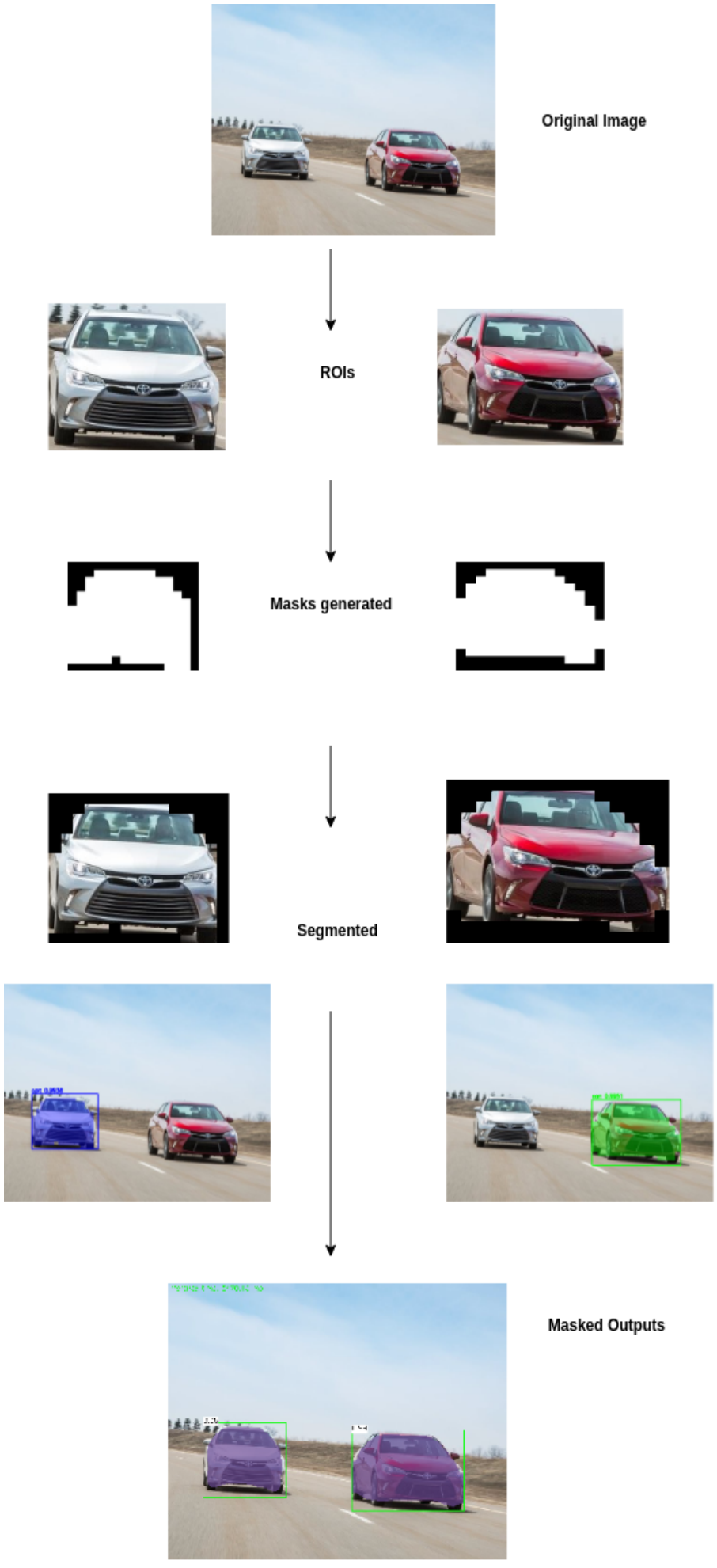


Figure 1: Application of Mask RCNN on the images of cars.

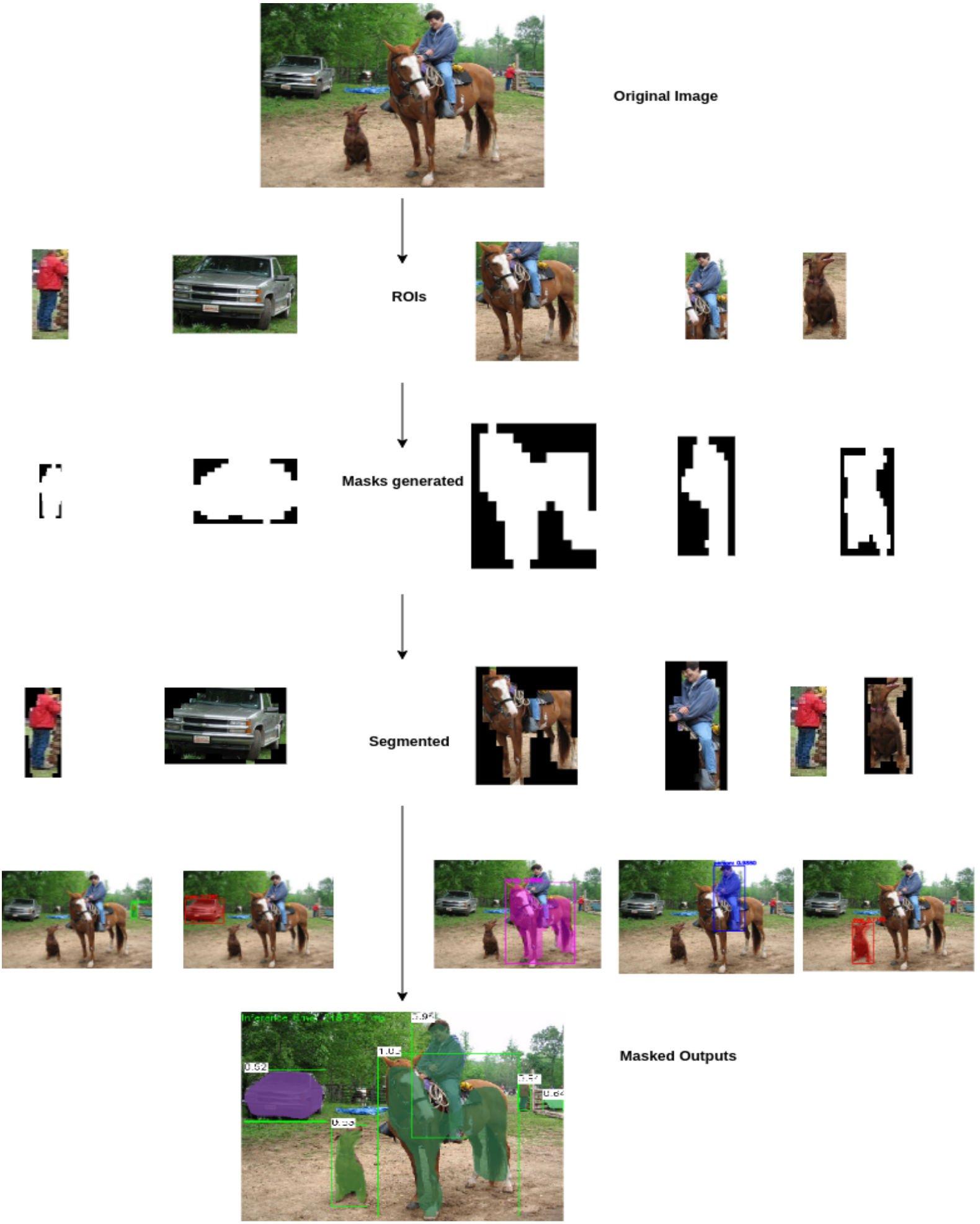


Figure 2: Application of Mask RCNN on the classic image containing horse, person, car, man and a dog.

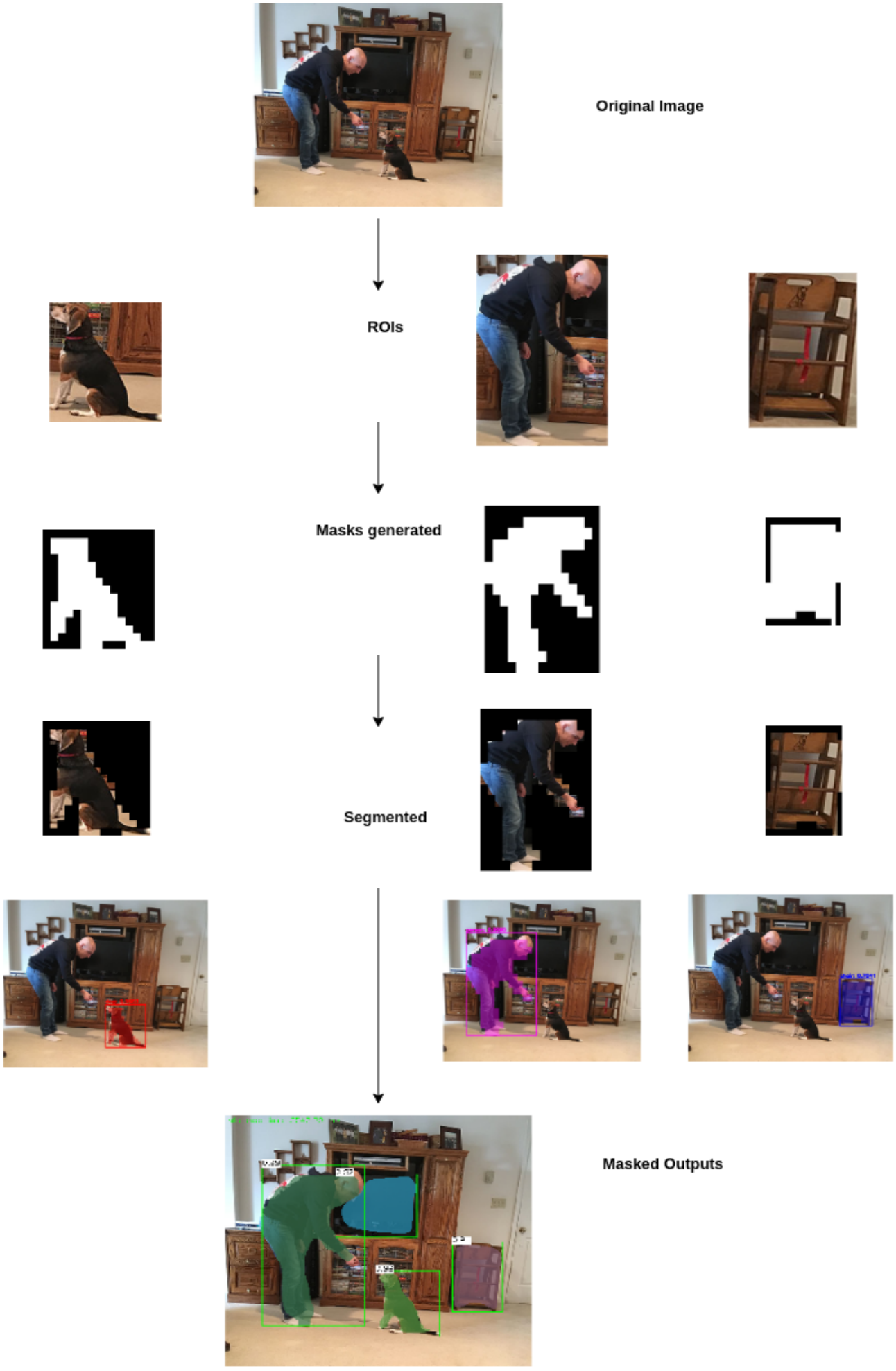
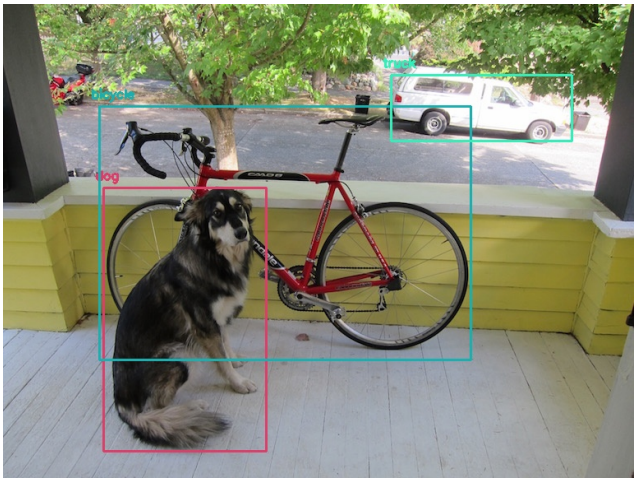


Figure 3: Application of Mask RCNN on image of Ardian and his dog.

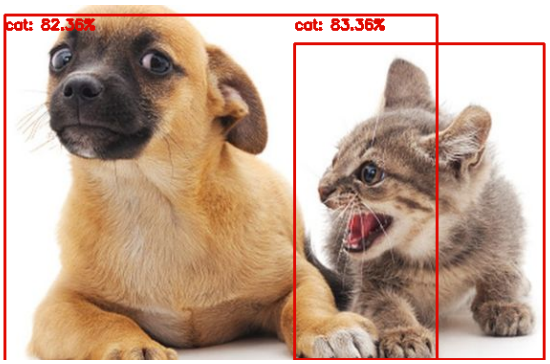
(a) The original image of bicycle, dog and car from Imagenet.



(b) The detected image of bicycle, dog and car from Imagenet.

Figure 4: Image of bicycle, dog and car from Imagenet, detected by the algorithm.

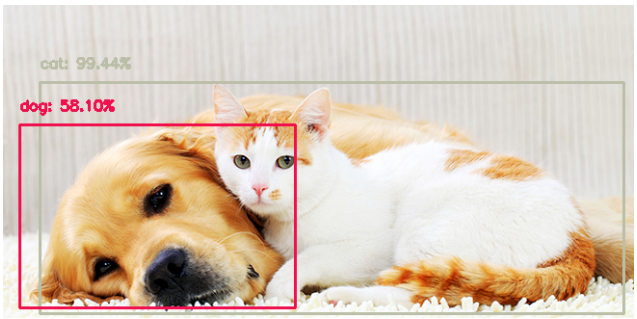
(a) The original image of cat and dog.



(b) The detected image of cat and dog with bounding box.

Figure 5: Image of a cat and a dog detected by the algorithm.

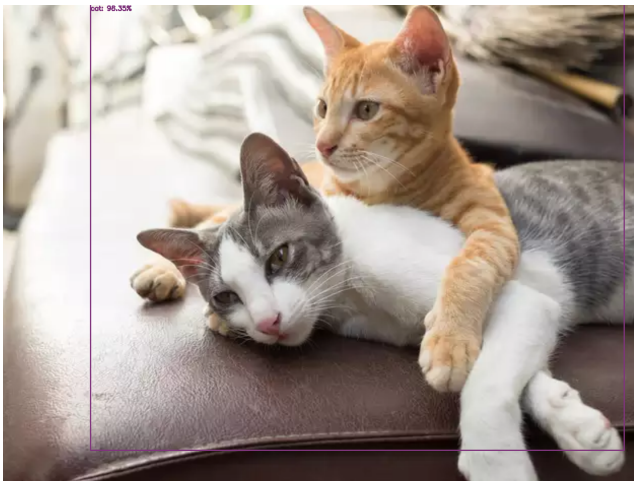
(a) The original image of cat and dog.



(b) The detected image of cat and dog with bounding box.

Figure 6: Image of a cat and a dog detected by the algorithm.

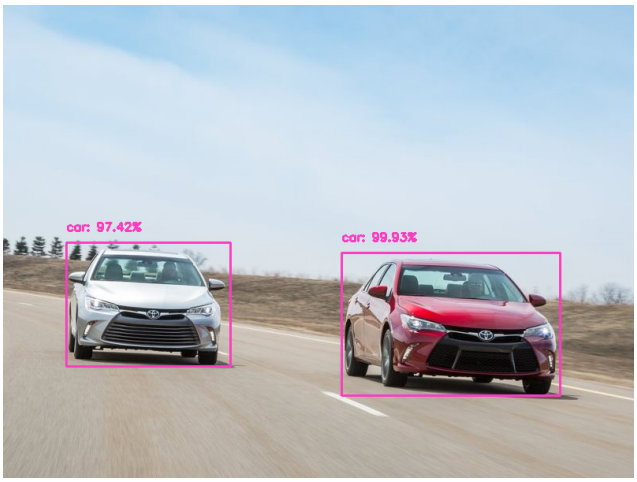
(a) The original image of cats.



(b) The detected image of cats with bounding box.

Figure 7: Image of cats detected by the algorithm.

(a) The original image of cars.



(b) The detected image of cars with bounding box.

Figure 8: Image of cars detected by the algorithm.

(a) The original image of plane.

(b) The detected image of plane with bounding box.

Figure 9: Image of plane detected by the algorithm.

(a) The original image of a horse jumping a hurdle.

(b) The detected image of a horse jumping a hurdle.

Figure 10: Image of a horse jumping a hurdle, detected by the algorithm.