Algorithms and Applications in Computer Vision – 046746

Homework 2 Report

Bar Goldner [redacted]

Daria Hasin [redacted]

14/12/2022

Part 1 – Classic Vs. Deep Learning-based Semantic Segmentation:

1. We loaded the images of the frog and horse folders:

Frog 1

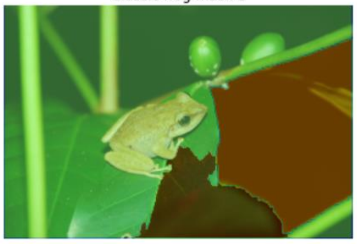
Frog 2

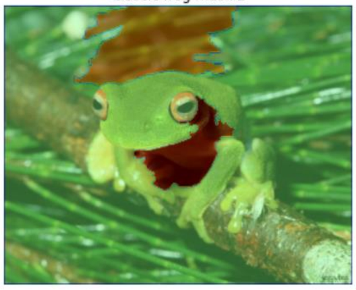
Horse 1

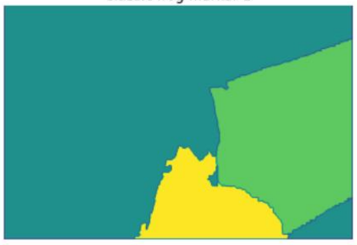
Horse 2

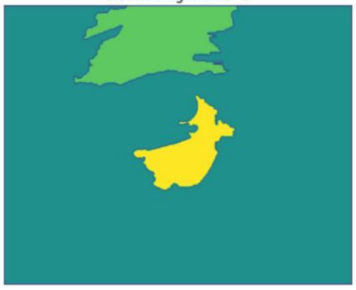
2. The classic method we chose for segmentation is **watershed**, this method operates like a topographic map on a grayscale image. The brightness of the image represents its height - high intensity represents the peaks and low intensity represents the valleys. It uses those heights to compute the gradients in iterations and then the segmentation.
   Advantages: low computation time, provides close contours, able to separate regions.
   Disadvantages: over-segmentation and under-segmentation.

   The deep learning-based method we chose for segmentation is **deeplabv3**. This method was developed by Google and with this network, they introduced atrous convolutions. With atrous convolution, we can modify the density of the computed features in the network and incorporate large context.
   Advantages: provides accurate segmentation for trained objects (ImageNet classes).
   Disadvantages: high computation time and under-segmentation of untrained objects.
   Segmentation with the classic method:

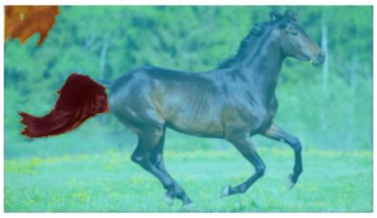

Classic frog mask 1


Classic frog mask 2

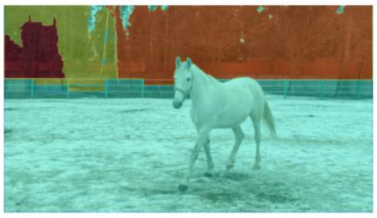

Classic frog marker 1


Classic frog marker 2

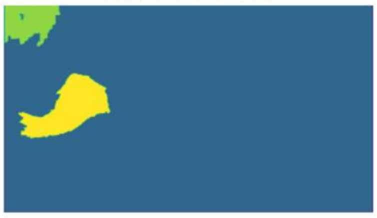
Classic horse mask 1



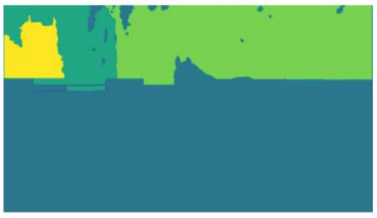
Classic horse mask 2



Classic horse marker 1

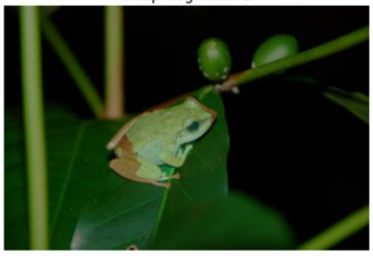


Classic horse marker 2

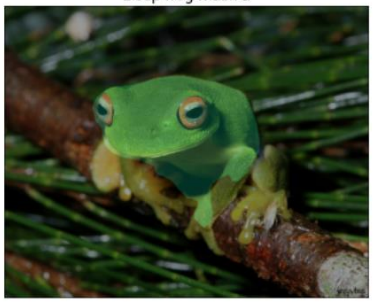


<u>Segmentation with the deep learning-based method:</u>

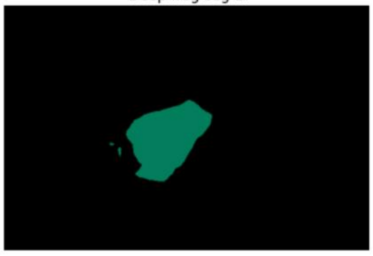
Deep frog mask 1



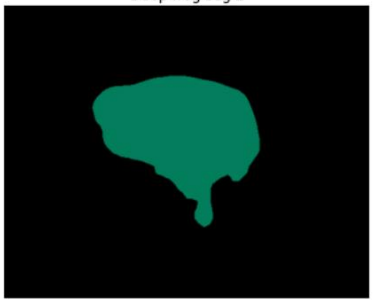
Deep frog mask 2



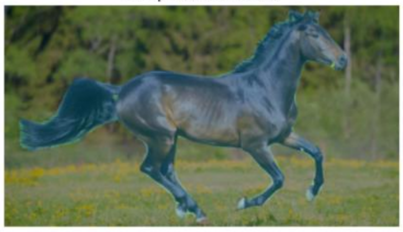
Deep frog seg 1



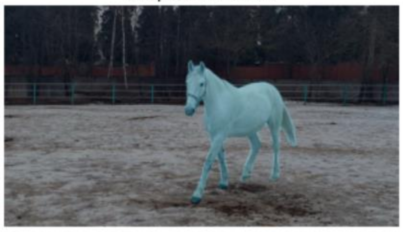
Deep frog seg 2



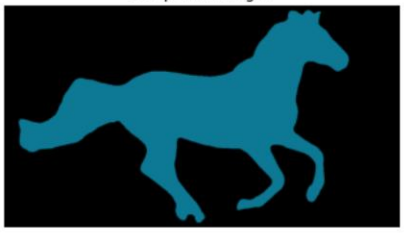
Deep horse mask 1



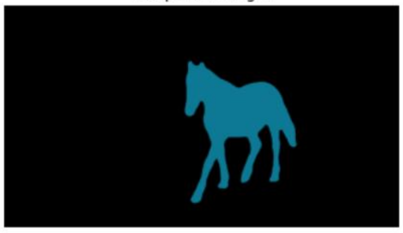
Deep horse mask 2



Deep horse seg 1



Deep horse seg 2



We made some pre-processing steps for the deep learning-based method; we converted the image to tensor (normalizes the values from 0 to 1) and normalized them for these parameters: mean= [0.485, 0.456, 0.406], std= [0.229, 0.224, 0.225] that were suggested in the documentation. We also tried to change some parameters in the watershed function and add pre-processing steps, but it didn't get better results

We can see in the images that the classic method's segmentation couldn't identify the objects clearly. On the other hand, with the deep learning-based method the objects were indeed separated from the background, although the segmentation of the frogs' images was not perfect. On the other hand, we can see that the segmentation of the horse is in the correct form.

3. The images that we chose were image of Eevee (Bar's Dog), image of cup of tea and image of diamonds (a girl's best friend):

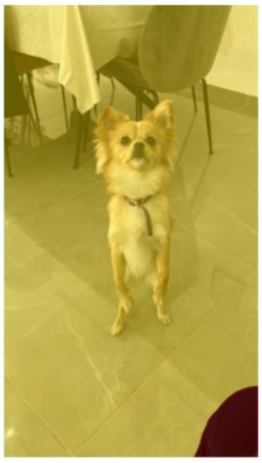
Image 1

Image 2

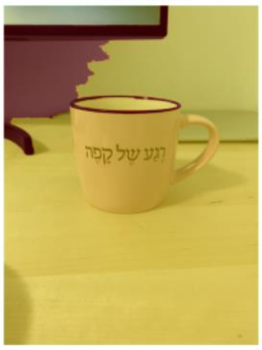
Image 3

4. We used the same classic and deep learning-based methods from section 2.
<u>The results of the classis segmentation:</u>

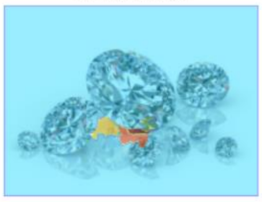
Classic mask 1

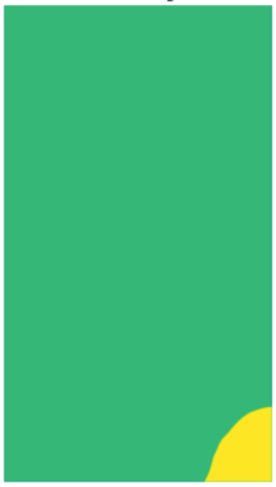
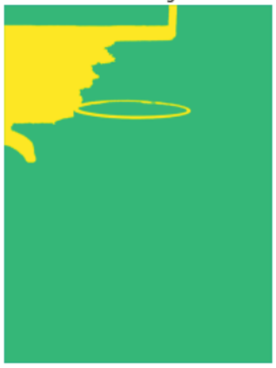
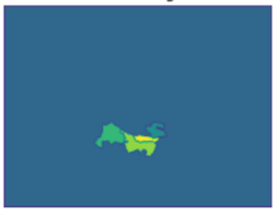


Classic mask 2

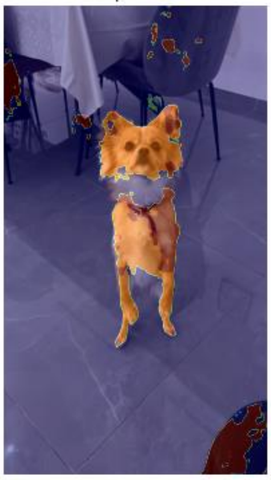
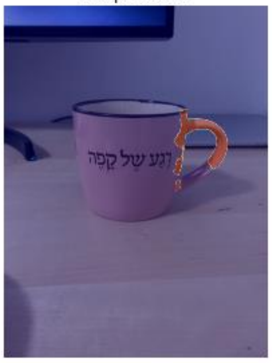
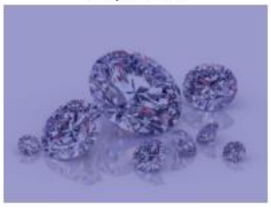


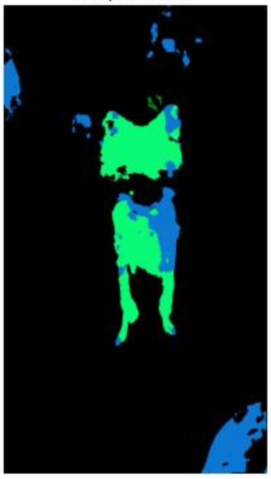
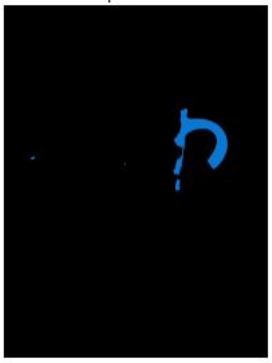
Classic mask 3



Classic seg 1

Classic seg 2

Classic seg 3


The results of the deep learning-based segmentation:

Deep mask 1

Deep mask 2

Deep mask 3


Deep marker 1

Deep marker 2

Deep marker 3

We can see that DeepLabV3 performed better on Eevee's photo than on the other photos. We assume it is the result of the training photos the network got from the dataset and ImageNet classes. The other two photos are not in the list of classes the course faculty provided us, so we assume that the network didn't train on those objects.

The Watershed algorithm performed poorly and didn’t segment the objects in the photos.

5. We noticed that segmentation can be rough around the edges. Some suggestions for alleviating the problem:
    - Pre-processing: smoothing the image so the edges will be continuous.
    - Inside the algorithm: check in the probability map if the second-best class for a pixel is the same as the nearest pixels, if it is – change it.
    - Post- processing: use morphological transformations for closing holes -> dilation followed by erosion.

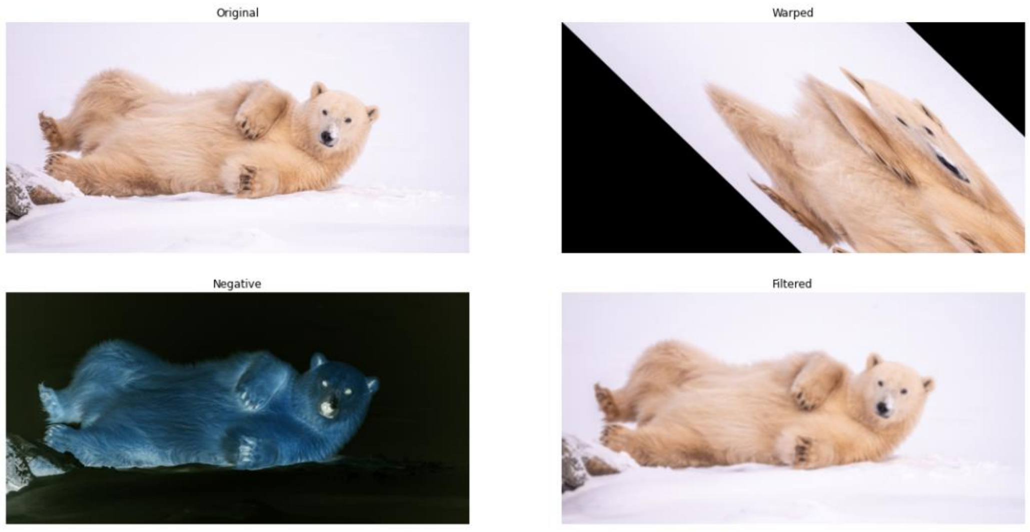
Part 2 - Adversarial Images:

1. We loaded the pre-train classifier Vision Transformer - vit_b_32.
   The Vision Transformer is a model for image classification that employs a Transformer-like architecture over patches of the image. An image is split into fixed-size patches, each of them are then linearly embedded, position embeddings are added, and the resulting sequence of vectors is fed to a standard Transformer encoder. In order to perform classification, the standard approach of adding an extra learnable "classification token" to the sequence is used.
   It pre-trained on large amounts of data and transferred to multiple mid-sized or small image recognition benchmarks (ImageNet, CIFAR-100, VTAB, etc.).
   Vision Transformer attains excellent results compared to state-of-the-art convolutional networks while requiring substantially fewer computational resources to train.
2. We chose an image of a polar bear in the snow that we found online and saved it to 'my_data' folder. We convert the image to RGB color and display it:

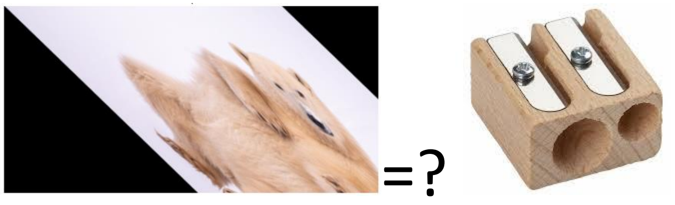
   We did a conversion from class index to label, with the supplied txt file and feedforward the image to the pre-trained network. The network's prediction was indeed polar bear:

```
Prediction:  ice bear, polar bear, Ursus Maritimus, Thalarctos maritimus
```

3. To create new images of a polar bear with transformations, we applied warping transformation using an open-cv affine transformation function. We also applied negative transformation by taking the value of each pixel and subtracting from it 255 in order to get a negative pixel value. Finally, we applied blur filter in size of 7x7 and sigma=20. The new images:



4. We fed the transformed images to network and the output results were surprising, we thought that the network will not recognize only the 'Negative' image, but the 'Negative' and 'Filter' images were recognized as polar bear while the 'Warped' image was recognized as a pencil sharpener.

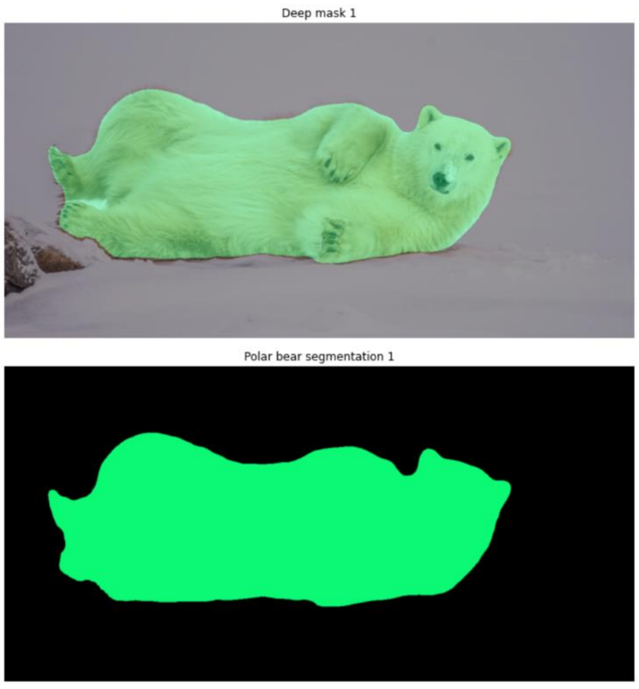


Okay, we kind of understand why it recognized it in that way...
The results:

```
Warped:
Prediction:  pencil sharpener
-----------------------------------------------------------------
Negative:
Prediction:  ice bear, polar bear, Ursus Maritimus, Thalarctos maritimus
-----------------------------------------------------------------
Filtered:
Prediction:  ice bear, polar bear, Ursus Maritimus, Thalarctos maritimus
```

As we mention, in this model, the image is split into fixed-size patches. So, when we warped the image, the small patches changed, and the model apparently detected patches that were in the images of sharpeners.
This contrasts with the blur filter and color change that did not change the positions of shapes and edges in the image. Apparently, the color change did not affect the identification because of the architecture of the model that could deal with this constant change (every pixel was changed in the same way as all pixels).

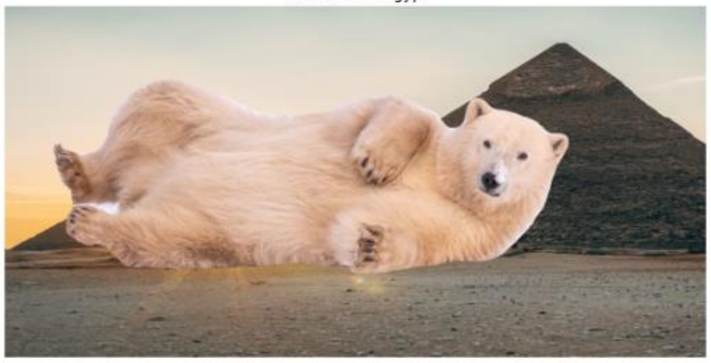
5. We used the same deep learning-based method from part 1, section 2 and segmented the image of the polar bear.
The result of the deep learning-based segmentation:



The segmentation succeeded in separating the polar bear from the image.

6. In order to put the polar bear in a different habitat, we used the segmentation mask in a way that we copied the polar bear image and for the pixels that are 0 at the segmentation image, we put the background image. In this way, we replaced only the background of the polar bear image because the pixels value of the segmented area is different from 0.
   For the background we chose an image of the pyramids in Egypt that we found online and save it to 'my_data' folder, this background is a different habitat for the polar bear.
   The result:

   Polar bear in Egypt

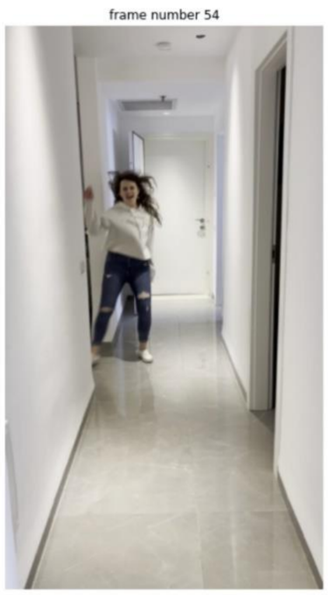
   The result image was saved to the 'output' folder.
   We wanted to make the homework-checker laugh but we thought that the result was kind of sad due to the global warming 😮
7. We fed the new image to the classifier and the prediction was conch:

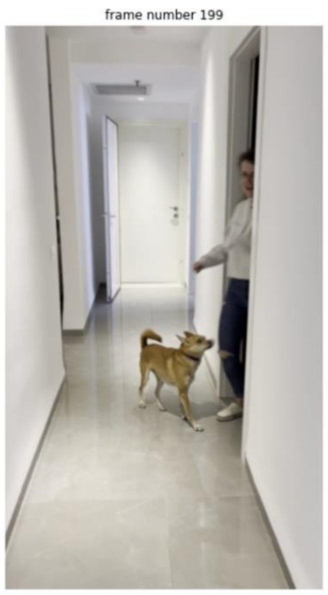
```
Prediction:  conch
```

   The prediction was different from section 2, because again - as we mention, in this model, the image is split into fixed-size patches. That is why when we changed the image background, we changed the small patches of it to patched that contains a golden-white object that is found on a background of sand. The conch is a sea animal that lives in its spiral shell, so images of conch could have patches with a golden-white object that is found on a background of sand.
   We think that the model detected similar patches between the images and that is the reason why it happened.
   For the record, we can understand the confusion:

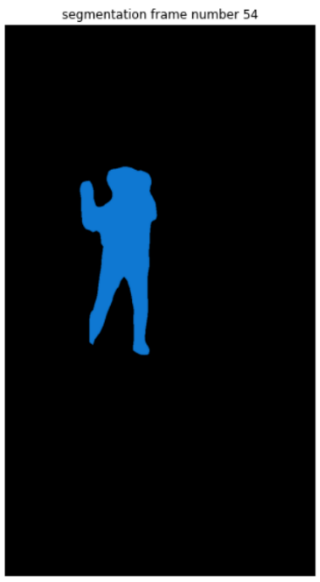
<u>Part 3 - Jurrasic Fishbach:</u>

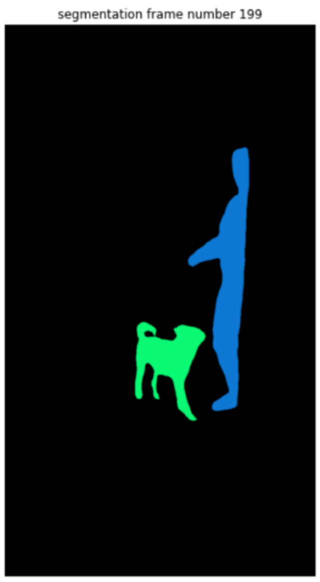
1. We filmed a short video of 8 seconds of Daria (and Bony – Bar's dog who wanted to join the party). We converted the video to frames by using the given function in the 'frame_video_convert.py' file and resized the frames from 1920x1080x3 to 1280x720x3. Two frames from the video:



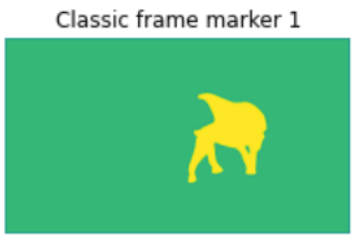


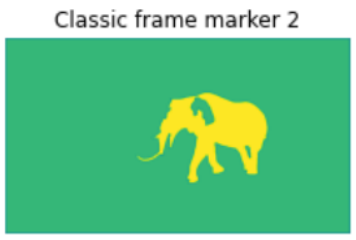
2. We segmented Daria and Bony (the dog) from the video using DeepLabV3:



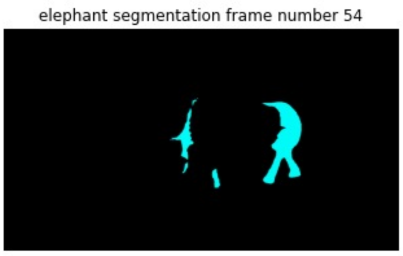


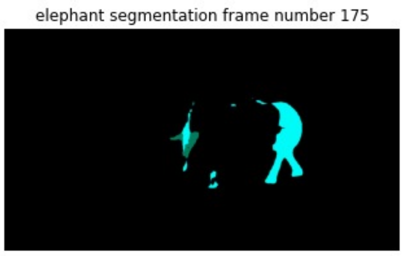
3. First, we chose green-screen video of an elephant, and we fed the frames to the classic method from part 1 section 2 and got a segmentation that is not accurate within the boundaries of the elephant:



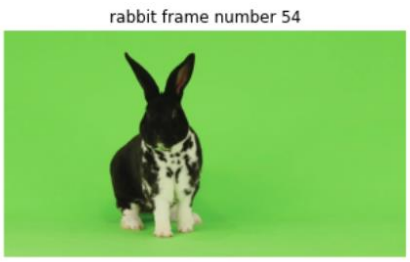


Then, we fed the frames to the deep learning-based method from part 1 section 2, which we made sure that elephant does appear as a label! But also, we got a segmentation that is not accurate within the boundaries of the elephant:

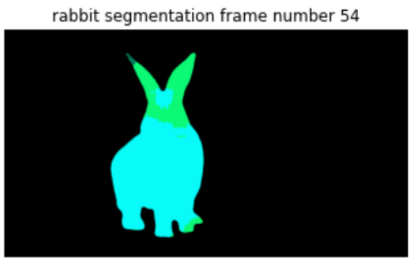


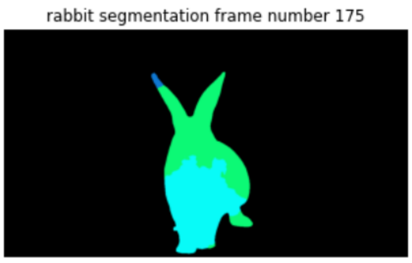


(We admit that we didn’t try to combine both classic and deep learning-based method). We wanted the video to be perfect, so we changed the green-screen video to a rabbit:



This video had a good boundary segmentation in the deep learning-based method, we again made sure that rabbit does appear as a label in this model:





We can see that the bunny got some classes, but it doesn’t matter for our purpose.

4. We wanted the video to be in the right proportion, so we made some pre-processing steps on the photos before combining them together:
   - Background photo – we reflected the bottom border of the photo by an offset of 55px and cropped it to be in the size of our video (1280x720).
   - Bunny frames and segmentation – first, we flipped the image because we wanted the bunny to walk on the ceiling. Second, we padded the images with zeros at the bottom by an offset of 2160px and at the right by an offset of 500px because we wanted to make the bunny smaller. Lastly, we resized the frames and segmentation to be in the size of our video (1280x720).

Then, we took the background image and stitched (frame-by-frame) Daria, Bony and the bunny to it after pre-processing.

We put it all together!
Our video is called “**Daria, Bony and bunny**” and it is in the output folder.
Enjoy! 😊

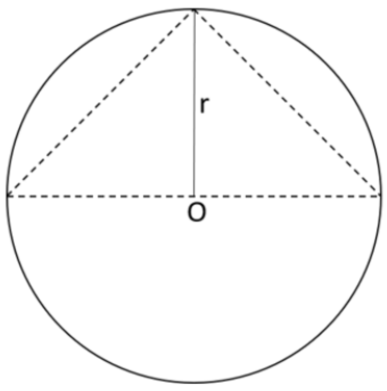
Part 4 - Dry Questions:

1. The difference between ordinary segmentation and semantic segmentation is that semantic segmentation is an algorithm that gives a label to every pixel in the image (e.g., Fully Convolutional Network). On the other hand, ordinary segmentation does not associate with a label for every pixel, it just portions the image into multiple segments (e.g., k-means clustering algorithm).

2. Calculation of the Intersection Over Union (IoU) in each where 'o' indicates the center of the circle, and the prediction of the algorithm is marked with a dashed line:
   As we learned,

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$

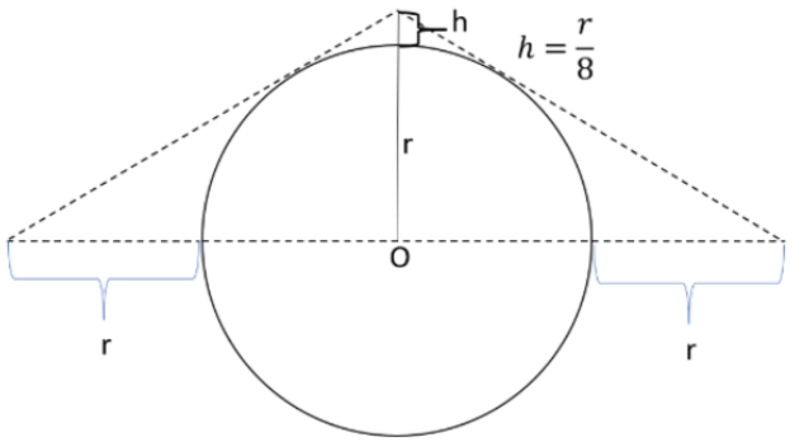
<u>Image A:</u>



$$IoU_{ImageA} = \frac{The\ entire\ area\ of\ the\ dotted\ triangle}{The\ area\ of\ the\ entire\ area\ of\ the\ circle}$$

$$IoU_{ImageA} = \frac{r^2}{\pi \cdot r^2} = \frac{\mathbf{1}}{\boldsymbol{\pi}} = \mathbf{0.318}$$

<u>Image B:</u>



$$IoU_{ImageB} = \frac{The\ area\ of\ the\ upper\ half\ circle}{The\ entire\ area\ of\ the\ triangle + the\ area\ of\ the\ lower\ half\ circle}$$

$$IoU_{ImageB} = \frac{\frac{\pi \cdot r^2}{2}}{\frac{4 \cdot r \cdot (r+h)}{2} + \frac{\pi \cdot r^2}{2}} = \frac{\frac{\pi \cdot r^2}{2}}{\frac{4 \cdot r \cdot \frac{9r}{8}}{2} + \frac{\pi \cdot r^2}{2}}$$

$$IoU_{ImageB} = \frac{\pi \cdot r^2}{4.5 \cdot r^2 + \pi \cdot r^2} = \frac{\boldsymbol{\pi}}{\mathbf{4.5} + \boldsymbol{\pi}} = \mathbf{0.411}$$

We can see that the IoU of Image B is greater.