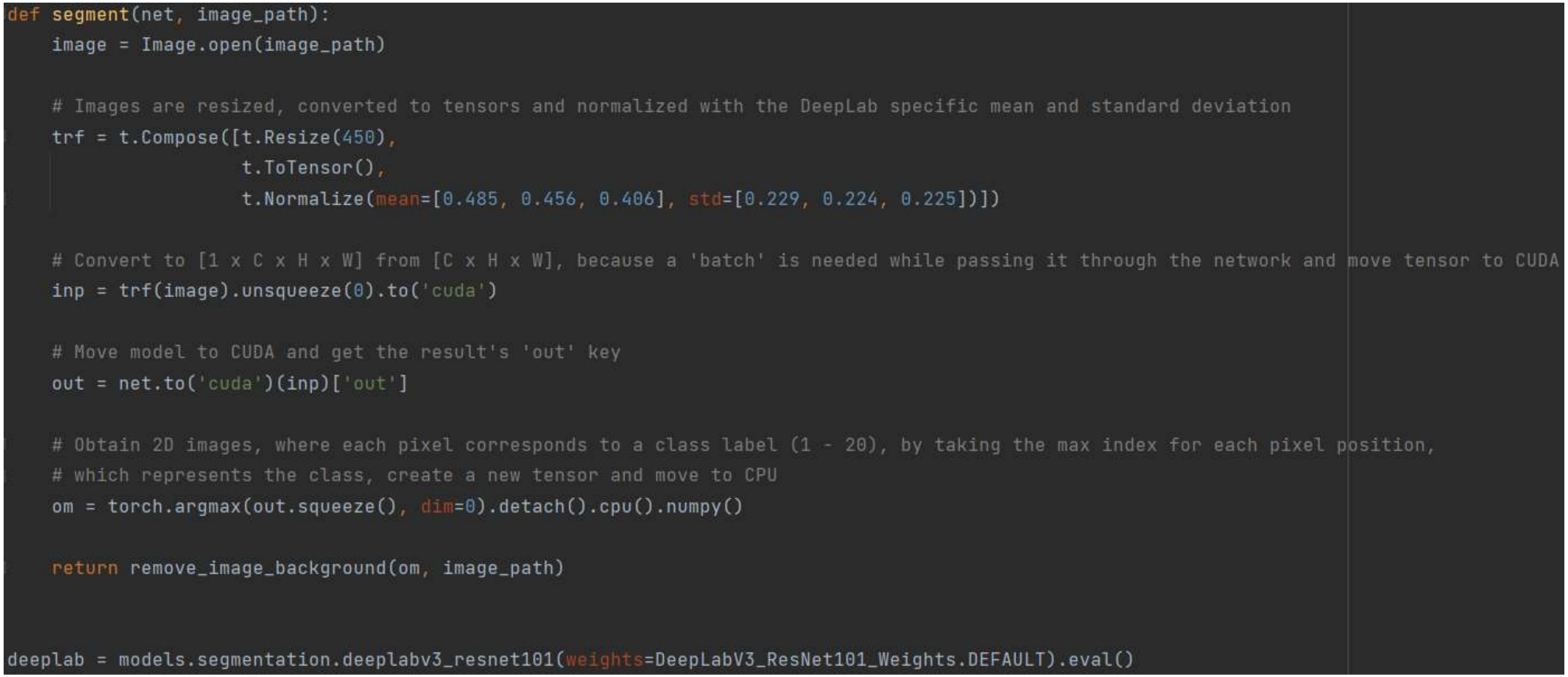
Photo Background Removal

Supervisor: Dr. Petru Radu

Student: Adrian-Ioan Tuns

Introduction

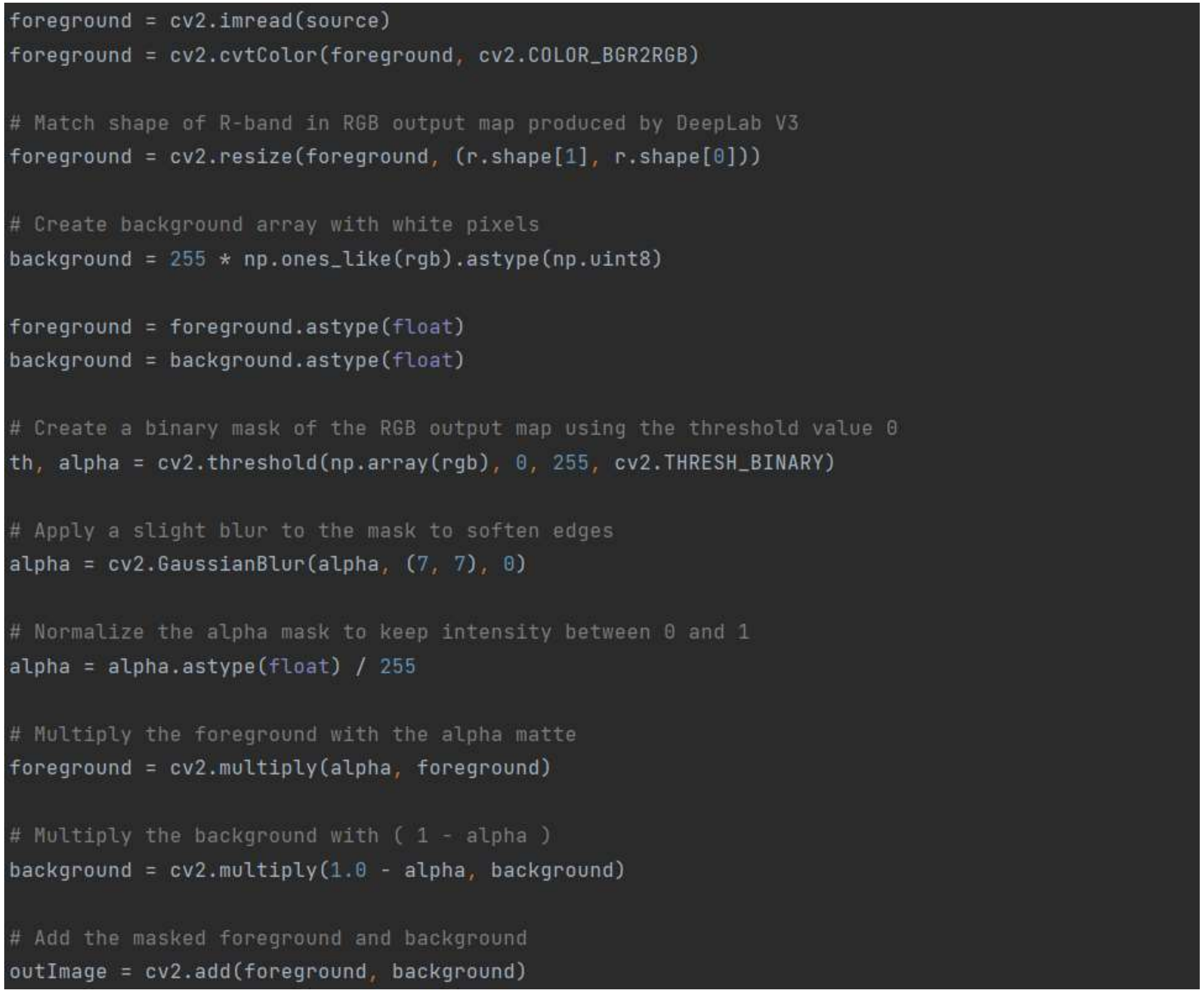
- The need for background removal
- Semantic Image segmentation
- DeepLab v3 from TensorFlow, Google
    - 20 categories supported, from objects to animals and humans



```python
def segment(net, image_path):
    image = Image.open(image_path)

    # Images are resized, converted to tensors and normalized with the DeepLab specific mean and standard deviation
    trf = t.Compose([t.Resize(450),
                     t.ToTensor(),
                     t.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])])

    # Convert to [1 x C x H x W] from [C x H x W], because a 'batch' is needed while passing it through the network and move tensor to CUDA
    inp = trf(image).unsqueeze(0).to('cuda')

    # Move model to CUDA and get the result's 'out' key
    out = net.to('cuda')(inp)['out']

    # Obtain 2D images, where each pixel corresponds to a class label (1 - 20), by taking the max index for each pixel position,
    # which represents the class, create a new tensor and move to CPU
    om = torch.argmax(out.squeeze(), dim=0).detach().cpu().numpy()

    return remove_image_background(om, image_path)


deeplab = models.segmentation.deeplabv3_resnet101(weights=DeepLabV3_ResNet101_Weights.DEFAULT).eval()
```

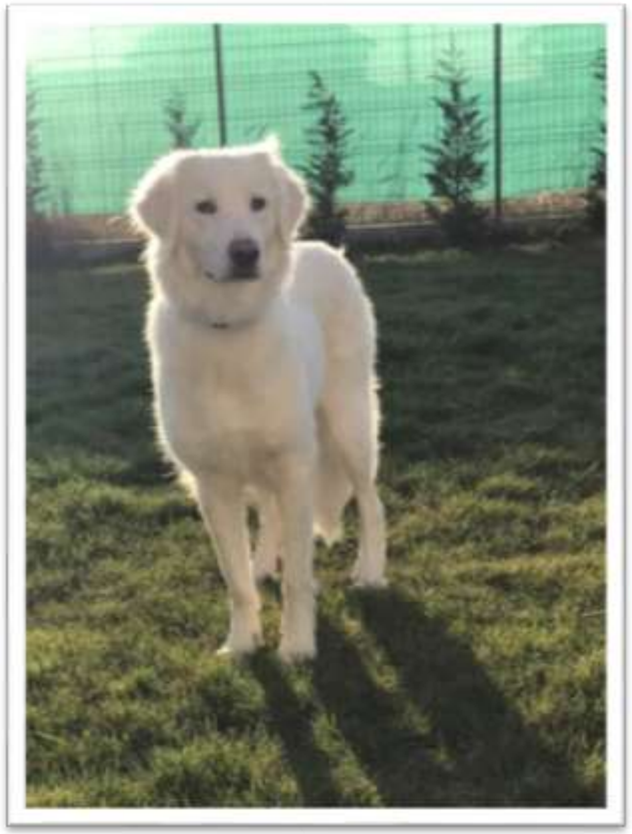
Background Removal

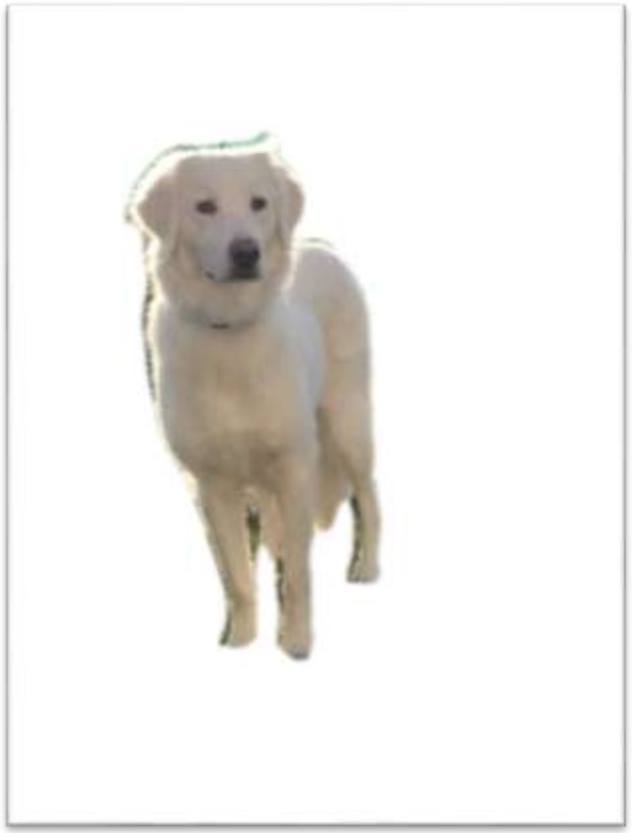
- Load foreground image
- Create white background image
- Binary thresholding is applied on the 3D rgb image obtained previously
  - The binary mask is stored in the “alpha” variable
- Blur alpha mask with GaussianBlur
- Alpha blending foreground with background



```python
foreground = cv2.imread(source)
foreground = cv2.cvtColor(foreground, cv2.COLOR_BGR2RGB)

# Match shape of R-band in RGB output map produced by DeepLab V3
foreground = cv2.resize(foreground, (r.shape[1], r.shape[0]))

# Create background array with white pixels
background = 255 * np.ones_like(rgb).astype(np.uint8)

foreground = foreground.astype(float)
background = background.astype(float)

# Create a binary mask of the RGB output map using the threshold value 0
th, alpha = cv2.threshold(np.array(rgb), 0, 255, cv2.THRESH_BINARY)

# Apply a slight blur to the mask to soften edges
alpha = cv2.GaussianBlur(alpha, (7, 7), 0)

# Normalize the alpha mask to keep intensity between 0 and 1
alpha = alpha.astype(float) / 255

# Multiply the foreground with the alpha matte
foreground = cv2.multiply(alpha, foreground)

# Multiply the background with ( 1 - alpha )
background = cv2.multiply(1.0 - alpha, background)

# Add the masked foreground and background
outImage = cv2.add(foreground, background)
```

Example of Background Removal





Thank you for your attention!