EECS 605: Technical Report of Virtual Background Project

Cheng Qian

April 15, 2022

This document contains the detailed information of my virtual background project, and instructions to reproduce this project on your own. In this project, I create a React app which enables the user input a human portrait and return an image with the background changed. The model is based on FCNResnet andis deployed on AWS.

1 Introduction

In the recent past, video calls have become a primary form of communication for work and school due to the global pandemic. With the closing of offices, schools, and studios, virtually everyone has been forced to use their homes as the centerpiece for their video calls. Many people are limited to the spaces with access to a computer and internet within their homes. These spaces can have distracting or unprofessional backgrounds for video calls. To minimize the effects this can have on people working virtually, they would like the ability to change the background of their video calls to an image of their choice that best fits the scenario or environment they are in.

Virtual backgrounds have been implemented in the past using the chroma keying method with green screens. This method replaces the pixels of a select color from an image or video with the desired background of choice [1]. However, this method will not work for most people as they do not have access to a green screen to use for their video calls to create a virtual background.

A better solution to implementing a virtual background is to use image segmentation. Image segmentation is a computer vision process which is a very mature technology. This technique partitions the pixels in an image or video frame into segments that represent a recognizable image object that can be classified. The classified pixels can be used to separate the object or person in the foreground from the background in a video call. Once separated, the background can be changed and composited with the foreground to create a virtual background effect. I have implemented this process using a neural network trained to classify objects in images. This technique doesn't need any reference and it enables people to replace the background at home in everyday settings with a fixed or handheld camera, instead of a studio.

I have read some resources and found that Resnet is a good choice for image segmentation. I found some paper which is really helpful for understanding Resnet [1][3]. Finally, I decided to use the fcnresnet101 pre-trained model as the base model and do transfer learning to do the segmentation task.

2 Approach

Residual Network (ResNet) models have proven themselves to be efficient and accurate classification models in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), being the 2015 Winner and 2016 runner up [3]. The ILSVRC is a yearly image classification competition where models are trained, validated, and tested using ImageNet's dataset with the winner being the model with the lowest error rate of mislabelling objects in images. The main idea of Resnet is introducing "identity shortcut connections" that skip one or more layers to overcome vanishing gradient problems [1].

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 1: Architecture of Resnet Variants [3]

Additionally, I selected the fcnresnet101 pretrained variant of the ResNet model as it can output a mask that classifies the pixels in an image. The architecture of fcnresnet101 is a Fully-Convolutional Network with the backbone of resnet 101. Resnet 101 is a neural network model with 101 layers. Table 1 shows the basic structure of layers for different backbones of Resnet. I choose the architecture with 101 layers. It has 100 convolutional layers followed by an average pooling layer and a 1000-class fc layer. The detailed description of this model can be found at: FCN ResNet-101 backbones.

The model is pre-trained using the COCO train2017 dataset with the 20 Pascal VOC dataset categories. The output of the net is an ordered dictionary and the "out" key contains the output tensors. The output is of shape (1, 21, H, W). 21 is the number of classes (including the "background" class). H and W represent the height and width of the original input. At each pixel, there are probabilities corresponding to the prediction of each class.

However, I aim to do virtual background of human portrait, the output of my model should only predict whether a pixel belongs to human. The output should have the size of (H, W). The value of each position means the possibility of belonging to human. Thus, I modify the pretrained model using transfer learning technique to tune the existing resnet model to fit the human portraits. I used pre-trained weights, removed last layers to compute representations of images. Then I added a binary classifier on top of it and trained the modified model from these features. The detailed tutorial of transfer learning can be found here: TRANSFER LEARNING TUTORIAL. All these modifying and training steps were done on my local machine.

The dataset I used for transfer learning and testing is a subset of images and videos from the Matting Human Dataset developed by AISegment. This dataset provides a diverse range of portrait images. For the purposes of image segmentation focused on people as subjects, this dataset was particularly useful.

3 Experiments

Using images and videos from the Matting Human Dataset, initial testing of my implementation began with the calculation of a segmentation mask with the use of the model. This segmentation mask served the particular purpose of identifying and classifying the person in the image. The original image and the mask is shown in Figure 2.

Then, once the segmentation mask is obtained, it is applied to the original image to extract the pixels representing the person. The inverse of the segmentation is applied to the background image to extract the background.

Finally, once both images are obtained, they are simply added together, therefore compositing

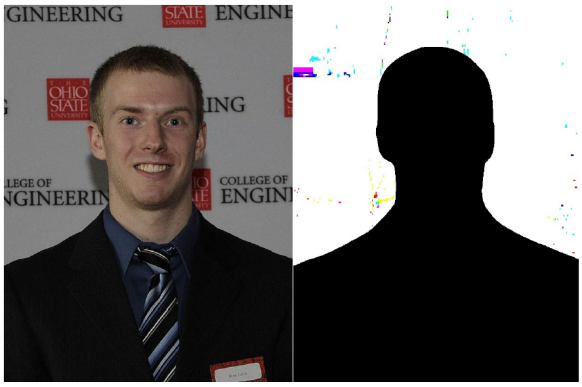

Figure 2: Original Image (left) and Segmentation Mask (right)

the foreground and background as intended. This implementation, as demonstrated on the following image, can also be applied to video data, where the process of segmentation, classification, and composition is done for each frame.

For the implementation, I first feed the images into the model. I get the mask for image segmentation from the output by setting a threshold. Then, I resize the input background to make it have the same size as the image with foreground. After this, I composite the foreground and background together by applying the mask and its inverse to the foreground and background, respectively, then adding the results.The final composited image is output as a JPG.

The evaluation of this model on the COCO 2017 validation set yielded a mean IOU of 63.7% and a global pixel-wise accuracy of 91.9%. The model on the test set has a pixel-wise accuracy of 90.2%. The loss function here is cross-entropy to determine whether a pixel is correctly classified.

4 Result

Figure 3 is an example which my algorithm works well. The left image is the input. The output perfectly replaced the background of the input.

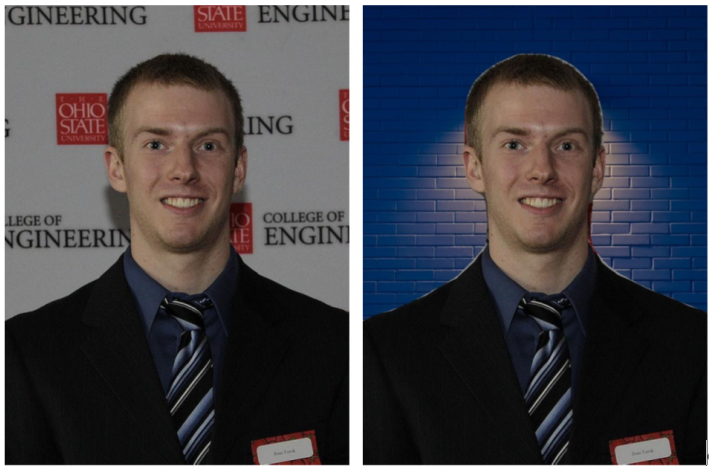

Figure 3: Example that Works Well

Figure 4 is an example which my algorithm doesn't work well. The model does not do a good job in extracting the outline of the character, especially the hair and the clothes. This may be because the action of the person in the picture is special, and the model has not seen this action in the training set. The model has poor performance on some other human portraits with special clothes or actions.

This may require people to sit upright in the center of the image, or increase the size of the training set.

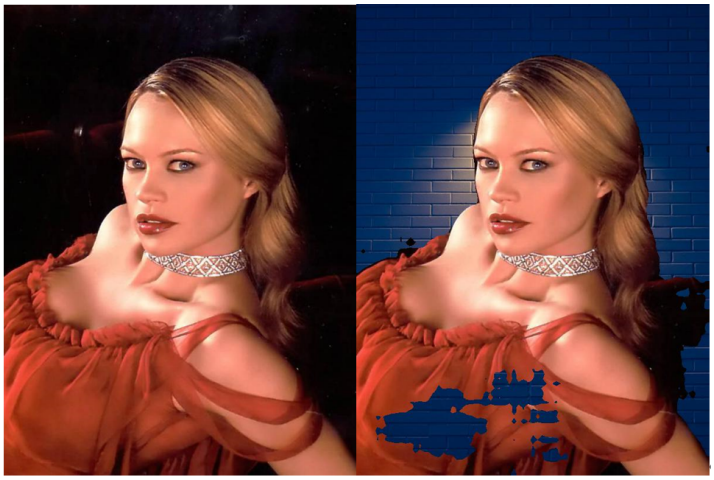

Figure 4: Example that doesn't Work Well

5 Cloud Architecture

The deployment and cloud architecture are based on AWS.

- Create a S3 bucket to save the model and default background images. The model is pre-trained on local machine and no training is needed on cloud.
- Create a new Lambda function in AWS which is responsible for pre-procesisng the input image and feed it to the model to get the output image.
- Another lambda function is created to handle the dropdown menu for demo on the website. It has 2 methods: GET and Post.
- Setup and deploy the API of the lambda function. This is usually how websites expose their services to public.
- Deploy the whole project on Heroku website through Github, so that the user can use the virtual background function on website.

Reference

[1] V. Feng, "An Overview of ResNet and its Variants," Medium, 17-Jul-2017. [Online]. Available: https://towardsdatascience.com/an-overview-of-resnet-and-itsvariants- 5281e2f56035.

[2] S. Kench, "What is Chroma Keying and How Does it Work?," StudioBinder, 05-Apr-2021. [Online]. Available: https://www.studiobinder.com/blog/what-is-chroma-keygreen- screen/.

[3] S.-H. Tsang, "Review: ResNet - Winner of ILSVRC 2015 (Image Classification, Localization, Detection)," Medium, 20- Mar-2019. [Online]. Available: https://towardsdatascience.com/review-resnet-winner-ofilsvrc- 2015-image-classification-localization-detectione39402bfa5d8.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385,2015.

[5] PyTorch. [Online]. Available: https://pytorch.org/hub/pytorchvisionfcnresnet101/.

[6] L. H., "AISegment.com - Matting Human Datasets," 06-Jun- 2019. [Online]. Available: https://www.kaggle.com/laurentmih/aisegmentcom-mattinghuman- datasets. [Accessed: 26-Apr-2021].