Photo/Video Effect Creation on a Person Using Semantic Segmentation

Marieta Baghdasaryan[1*], Nane Arshakyan[2*], and Vahagn Hakobyan[3**]

[***]Capstone Supervisor: Marianna Ohanyan
[*]*Bachelor of Science in Computer Science, American University of Armenia*
[**]*Bachelor of Science in Data Science, American University of Armenia*
[1]marieta_baghdasaryan@edu.aua.am
[2]nane_arshakyan@edu.aua.am
[3]vahagn_hakobyan@edu.aua.am

***Abstract*— Image segmentation has been widely applied in solving computer vision problems, from autonomous vehicles to medical imaging and photo effects. There are different approaches and techniques to image segmentation. However, each one has its specific applications in different use cases. In this work, we present video effects that are created using a semantic segmentation technique that is based on artificial neural networks. The effects include background replacement, RGB glitch, and person outline effect.**

***Keywords*— Semantic segmentation, convolutional neural networks, U-Net**

I. Introduction

With increased number of Big Data applications in businesses, image processing techniques have become crucial for organizing, managing, and making sense of image data. Computer vision is the field of Artificial Intelligence, which is responsible for deriving information from digital visual content. One way to extract information from images is image segmentation. It is a technique used to partition the pixels of the image into different regions and categorize those for further use (Shapiro and Stockman, 2000). Image segmentation applications vary from customer behavior tracking to video surveillance, traffic control systems, and health monitoring, including medical diagnosis, surgery planning, and autonomous driving (Vadapalli, 2021). Social media and other visual entertainment and content creation platforms also use image processing. An example of an image processing use case is the implementation of person detection and segmentation techniques for creating visual effects for photography and videography, which will be executed in this project. The widely popular ways of image segmentation are instance segmentation, semantic segmentation, and the recently proposed panoptic segmentation.

Semantic segmentation is a process that clusters all pixels into classes of an object. One example can classify all people in a photo into one segment and their background into another segment. The following process, instance segmentation, identifies and labels each pixel into individual instances of an object. For example, in Fig. 1a and 1b, you can observe an example of semantic segmentation and instance segmentation.

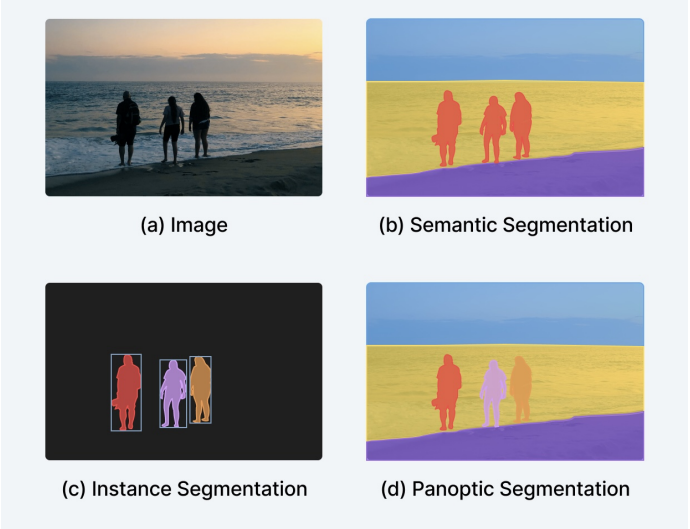

Fig. 1. Visual representation of segmentation approaches: a - the initial version of the image, b - semantic segmentation, c - instance segmentation, and d - panoptic segmentation (Barla, 2022).

The last approach of image segmentation is panoptic segmentation (Fig. 1d). Each pixel is assigned two labels throughout the segmentation process, their instance and semantic labels (Kirillov et al., 2019). In other words, panoptic segmentation is used to identify and categorize everything on an image, from single objects to uncountable things, like grass, road, etc. This process is mainly used for autonomous driving, medical imaging, and digital image processing.

The typical real-world applications of semantic segmentation have a profound impact across different industries. Semantic segmentation can be used for self-driving cars, retail, and medical scans (Keymakr, 2021). The technique can help an autonomous vehicle's orientation by identifying pedestrians, barriers, other vehicles, and lanes in the first case. On the other hand, the retail industry uses semantic segmentation to suggest to customers similar items based on their choice. This segmentation works perfectly well for digital imagery and photo editing, particularly distinguishing between humans

and their backgrounds. One popular use case of semantic segmentation in photo effects can be the portrait mode on the iOS cameras, which use photo segmentation mattes to distinguish between the background and the foreground of an image and thus blur the background (AppleInc., 2019). Throughout this project, the following effects will be created using semantic segmentation:

- Background replacement
- RGB glitch effect.
- Person outline

The code implementation of this project can be found here: https://github.com/NanehArsh/Capstone.git

II. Literature Review

There are two fundamental approaches when it comes to image segmentation. The first approach is similarity detection, and the other one is discontinuity detection. In other words, similarity detection tries to detect similar pixels into a group, combining separate pixels into a region, and the other approach identifies the edges of the regions first through algorithms such as edge detection or line detection. Core methods that are used to execute image segmentation are image processing and deep learning methods. The most common techniques used for image segmentation through these methods are the following:

- Threshold-based segmentation
- Edge-based segmentation
- Region-based segmentation
- Clustering-based segmentation
- Neural-networks-based segmentation

The first four methods represent image segmentation techniques, whereas neural-networks-based segmentation is done through deep learning methods.

A. Thresholding-based segmentation

The image's color space is converted into grayscale when applying this method (Al-amri et al., 2010). The segmentation is done by assigning a luminance intensity value to each pixel and dividing the image into two segments based on the threshold value. Although this segmentation technique is quite simple to implement, a thing to consider is that it has poor performance when it comes to the details; they are mostly left out.

B. Edge-based segmentation

Throughout this process, the edges of the objects are located on an image. The emphasis is on the edges, as they are regarded as the most significant pieces of information on an image (Yuheng and Hao, 2017). The image size is greatly reduced, and irrelevant information is eliminated from the image. In this case, the image is also being divided into binary segments, one including the pixels that are classified as edges and the other including the rest of the pixels. It classifies the pixels based on the discontinuity of contrasts, textures, luminance, saturation, etc. It is best used when the objects on an image have accentuated contrasts. However, the efficiency drops significantly in noisy images.

C. Region-based segmentation

This technique of segmentation is based on dividing the image into multiple regions that share similarities, selecting seed pixels for each one of those, and later merging or splitting further the pixel sets that are located around the same neighborhood of pixels, with a seed in the center (Yuheng and Hao, 2017). Region-based segmentation works pretty well even on noisy images; however, one major drawback is that it requires a lot of memory and time.

D. Clustering-based segmentation

Clustering-based segmentation locates the pixels with similar features and groups them into specific clusters (Sharma, 2020). These pixels do not have to be neighboring each other. Clustering images into color groups, for example, might help us detect objects that differ from each other by their color. This segmentation technique works pretty well on smaller datasets; however, the computation time might be too large for larger datasets.

E. Neural-networks-based segmentation

Neural network-based segmentation analyzes and identifies different objects in an image. It is done by training the data on an image dataset to recognize a particular pattern later (Sharif et al., 2010). The benefits of this technique are that it is easy to implement, Python libraries are available, and it has practical applications (Prasad, 2020). However, training such a model on big datasets might be time-consuming and costly.
Convolutional neural networks (CNNs) work pretty well with images that have three dimensions. The dimensions are the height, the width, and the RGB value intensity of the image (Kaushik and Kumar, 2019). Based on the benefits of neural network-based segmentation, the project will be executed by implementing a convolutional neural network with a U-Net architecture.

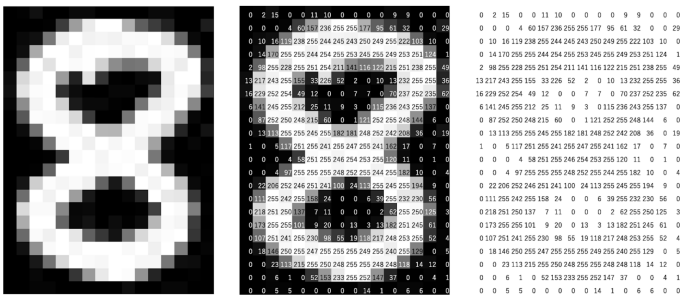

Fig. 2. The pixelated image on the left, the pixels with their corresponding pixel brightness value in the middle, and the matrix of those values on the right (Yamashita et al., 2018).

III. Prerequisites

The advances in convolutional neural networks help improve image recognition (Long et al., 2015). They can be used very effectively in semantic segmentation. The following section highlights the building blocks of CNN that will help to dive deep into the U-Net architecture.

Inspired by the neural networks found within the mammal brain, artificial neural networks consist of neurons and edges. Each neuron takes a signal, processes it, and transmits it to the next neuron via an edge. Each neuron and edge has a weight assigned to it. It signifies the value that each of those carries. Higher weight value corresponds to higher importance of the neuron. The weight, however, continuously changes to fit throughout the learning process of a model to ensure more accurate results. U-Net is such an architecture through which semantic segmentation can be executed (Zhang, 2019). It is a modification of a traditional convolutional neural network introduced to improve biomedical research. It can locate and identify objects in specific regions of an image. It consists of contractive and expansive paths that shape the U-like structure.

Using this architecture for semantic segmentation is useful, as it labels the pixels that already have their locations marked with their corresponding category value (e.g., person and background). A machine can identify an image by using an array of numbers. In an image with a depth of 8 bits, the value of each pixel can take values between 0 and 255 (Visual representation in Fig. 2). For each pixel, there are three color channels: the Red, Green, and Blue channels (RGB), each corresponding value between 0 and 255. The combination of these three channels makes up the colorful images. To represent an image in the form of a matrix, we need a 3-dimensional matrix, having the height and the width of an image as two dimensions and the depth (RGB channels separately) as a third dimension.

A CNN architecture consists of multiple building blocks, including convolutional layers and pooling layers.

A. Convolution and activation function

The first layers are crucial in the feature extracting process (Yamashita et al., 2018). It is a two-step process, where firstly, a convolution operation is performed, followed by an activation function. Convolution is a linear operation performed to extract features from an image. Throughout the process, a kernel, a small array of numbers, is applied through an input array of numbers called an input tensor. After this procedure, the feature map is created (Fig. 3). It can be observed that the outermost rows and columns are being left out of the process, and the information from border pixels is lost. However, a technique called padding can be used to address the issue. Through this process, rows and columns of zeros or mirrored values are added to the input tensor so that the values on the edges are not missed out.

Nonlinear activation functions are significant to ease the process of training (Ramachandran et al., 2017). It is being applied to the feature map after the convolution. Rectified Linear Activation function (ReLU) has been one of the most widely applied activation functions due to its effectiveness and simplicity. Each of the numeric pixel values in the feature map is redefined by the following function:

$$f(x) = max(0, x)$$

The function will directly output the input in case it is positive and 0 otherwise.

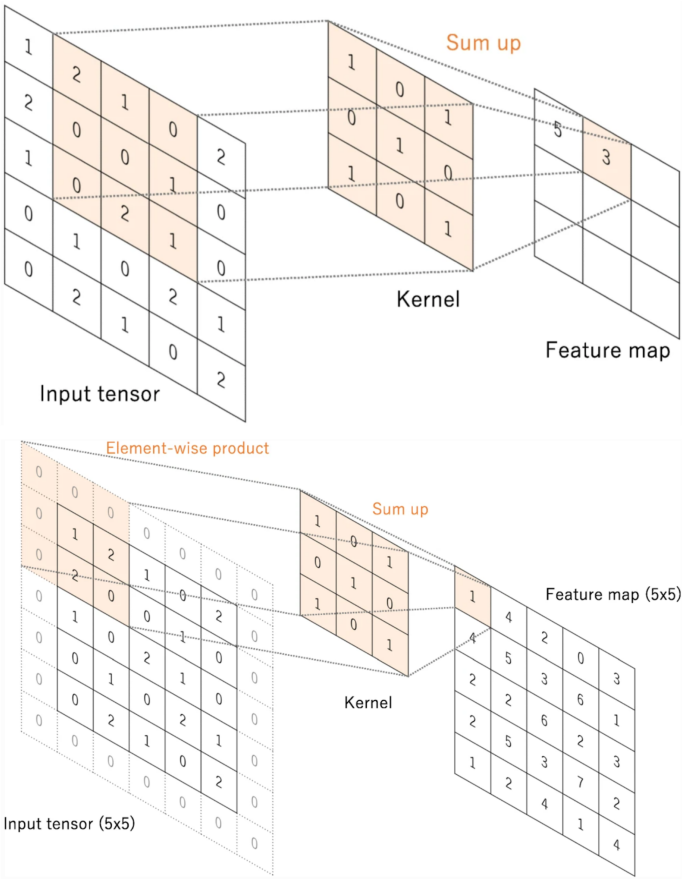

Fig. 3. An example of a convolution operation with a kernel size of 3×3, an input tensor with no padding on the top and the tensor with padding on the bottom (Yamashita et al., 2018).

B. Max pooling

Max pooling is a pooling operation where, depending on the length of a stride, the input feature map is being downsampled by the maximum value of each MxN group of numbers. All the other values in a patch are disregarded. The most common pooling filter would be a 2x2 filter with a stride of 2, that can be found in Fig. 4.

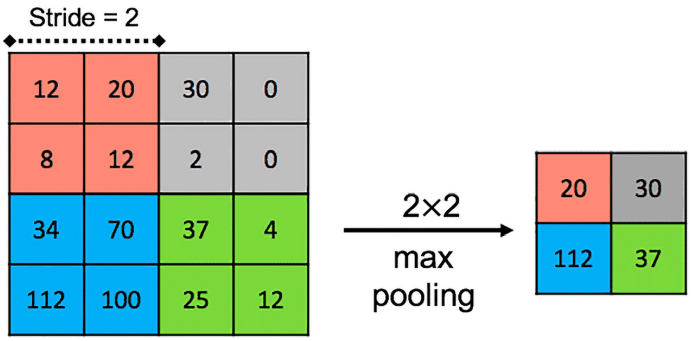

Fig. 4. An example demonstrating the process of max pooling (Gupta et al., 2018).

IV. Approach

This project aims to create photo/video effects for portrait imagery. The approach to problem-solving is divided into

two sections. Firstly the person is segmented out from the background; then, the obtained segmentation masks are used to create the effects. Convolutional Neural Networks, specifically CNN with the U-Net architecture, have been used to achieve the solution of the first sub-task.

A. U-Net architecture

In 2015, Olaf Ronnenberg, Philipp Fischer, and Thomas Brox published a report introducing a new and more efficient architecture for semantic segmentation. Although this convolutional network was initially aimed at biomedical applications, it has been widely used in other areas due to its training efficiency and results. The architecture is called U-Net, and an example is presented in Fig. 5. With this architecture, the process is divided into two parts: the contracting path on the left side and the expansive path on the right side of the image.

The contracting path follows a traditional convolutional network architecture. Firstly 3x3 unpadded convolutions are applied two times; a ReLU activation function is applied after each application. It can be observed that the image resolution decreases by two from both the height and the width after each convolution. It is due to the lack of padding. Next, a 2x2 max pooling operation with stride two is executed on the layer, thus downsampling the image. The feature channels are doubled after each downsampling procedure. In the bottom row of Fig. 5, where it is a 32x32 pixel image left with 1024 feature maps, up-convolution is applied to upscale those.

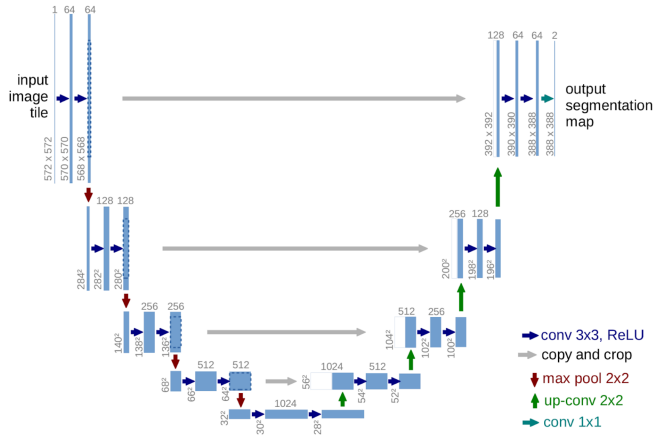

Fig. 5. U-Net architecture example. The blue boxes represent a multi-channel feature map. The white box represents a copied feature map. The arrows represent the convolution operations happening on different layers and the application of the ReLU activation function (Ronneberger et al., 2015).

The feature map channels are up-sampled on each up-convolution layer and are concatenated with the cropped feature maps from the equivalent line on the contracting path. Cropping is necessary due to the loss of border pixels on each convolution. The final stage of the expansive path is running a 1x1 convolution to map all the feature maps into multiple layers of classes (2 in the case of Fig. 5).

B. Effects

Multiple effects have been implemented on the images after the segmentation process. One significant advantage of person segmentation is that the effect can be applied separately to either background or the person.

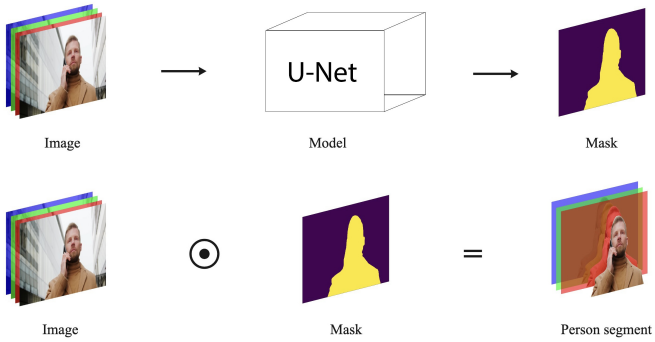

Fig. 6. Background removal: execution process.

1) Background removal: Person segmentation can effectively be used to manipulate the background of an image. The image is loaded and resized to go through the model. The resulted binary mask is multiplied with the original image to create a person segment later on, as presented in Fig. 6.

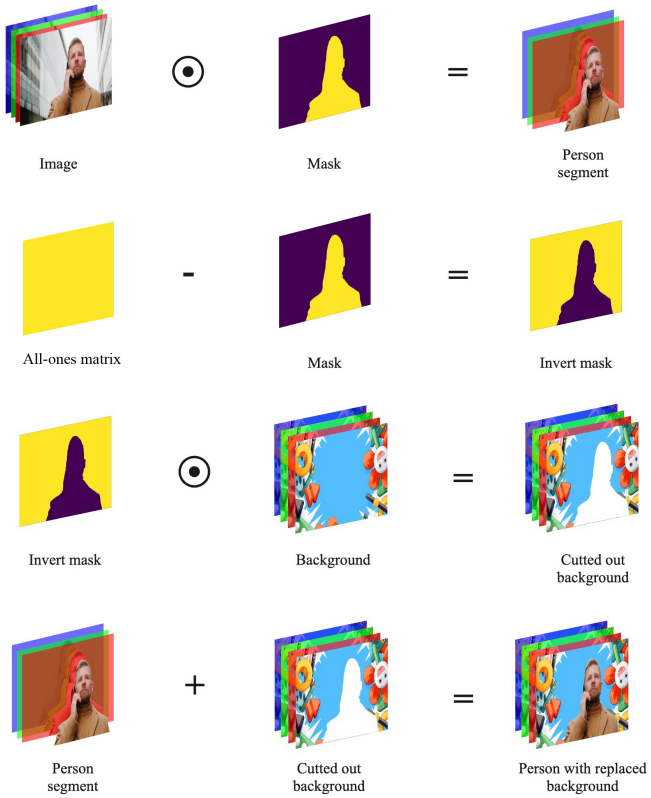

Fig. 7. Background replacement effect: execution process.

2) Background replacement: The background replacement takes the already segmented person as an array, and according

to the size of the input image, resizes the background image and automatically crops to fit in with the aspect ratio of the input image (Fig. 7).

3) RGB glitch effect: The following effect is created by splitting the image color channels into separate red, green and blue channels and moving those along the axis of the image's canvas (Fig. 9). After person segmentation, the output portrait image has been split into red, green, and blue channels. Padding has been added to two color channels only, 5% from the top and 5% from the left side, respectively. The padded channels have later been cropped to match the initial image resolution and finally merged back together.

4) Person outline: For this effect, the binary mask created by the model was taken, and applied to the blue channel of an image with with RGB channel with values of zeros (a plain black image with the resolution of the mask), an OpenCV function, contour detection, was applied to detect the edges between the person and the background, that will become the outline later (Fig. 8). The result of the function is a matrix with the position values of the outline pixels. These values are used to draw an outline on an image by replacing the pixels in the original photo.

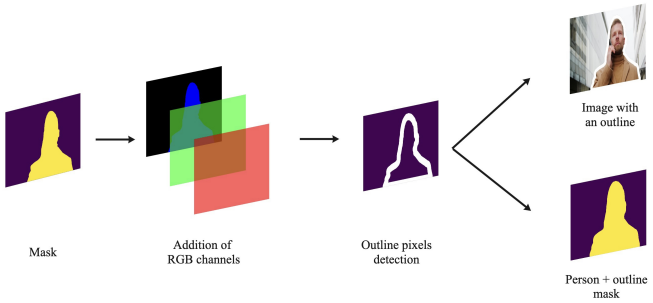

Fig. 8. Person Outline effect: execution process.

V. Model Training and Evaluation

The data has been collected from Kaggle, a platform to access public datasets and explore different models. The dataset consists of 18,500 samples of portrait images for training and 3,500 portraits for testing. Due to the small size of the images, 128x128 pixels, the training happened in a relatively short time. The image size has been selected to be 128x128, due to lack of computing power. Thus, the results are expected to be less accurate than those present in the source. Initially, the training and test sets have been prepared for training. The segmentation-models library has been used to access the predefined architecture of the U-Net model in python.

A Jaccard loss function have been defined to measure the efficiency of the model (Eq. 1).

$$Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

This function, also known as intersection-over-union (IoU) loss, evaluates the segmentation quality by dividing the intersection and the union of two sets, A and B. The values of the coefficient vary between 0 and 1. The former represents

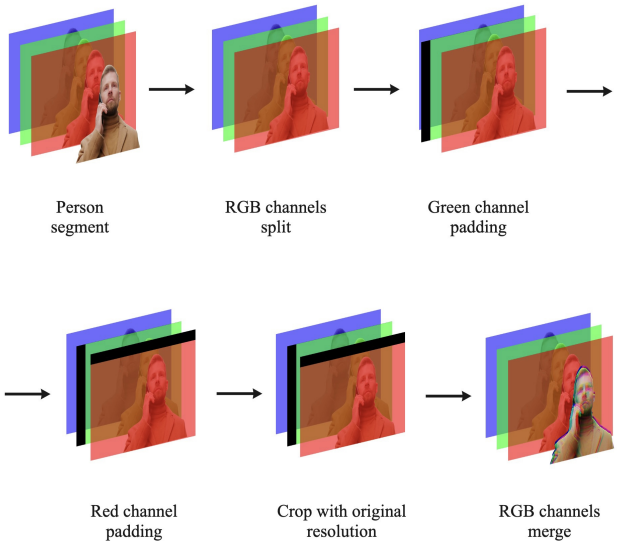

Fig. 9. RGB Glitch effect: execution process.

zero if the sets are disjoint, and it is one if they are identical (Duque-Arias et al., 2021). In our case, the intersection is the overlap area of the prediction and the ground truth image, and the union is the union between those two areas.

The Adaptive Moment Estimation Algorithm (ADAM) has been selected as an optimization algorithm. It is much more efficient than the gradient descent or stochastic gradient descent methods (Battini, 2018). It is easy to implement, does not require much memory, and works well with large datasets.

VI. Results

Three types of effects have been created through this project using semantic segmentation. The images that are presented in the following figures (Fig. 10, 11) are selected to be close-up portraits with neutral/mono-color background, and the results are close-to pleasing. The model training has been executed for five epochs with a batch size of 64 images. The IoU score after the fifth round of training has been 0.9648, meaning that the prediction mask is approximately 96.5% accurate. Compared to the results in the source of segmentation technique, however, we have got a lower IoU score: 0.9648 vs. 0.981. The difference might be because of the lower number of epochs run in our case. In the original study, the training was run for ten epochs. However, it is five within the scope of this project.

The model has been tested for images with multiple people on it as well; however, it performs better when only one person is on the image. Also, as the training dataset consisted of mostly close-up portraits, the model performs poorly when applied to the whole body. In addition, the model was run on the test set of the database to calculate the mean and the median time of the model execution per frame. The data was adjusted for outliers by keeping the values up to two standard deviations away from the mean. Table I was created by running the model on the test set of images of approximately 4000

images. According to our results, the mean execution time per frame was 0.077 seconds, and the median was 0.073 seconds. These numbers indicate that to run the model on 1 second of footage will take 2 seconds on average. If the frame rate is 25 frames per second (FPS), in case FPS is higher, more time will be required to execute a one-second video. Whenever the image resolution increases, we can observe a significant increase in the execution time of semantic segmentation per frame to approximately 0.15 seconds in the case of 256x256 images and around 0.54 seconds per 512x512 pixel resolution image. The results might improve if the model is trained on resolutions higher than 128x128 pixels.

TABLE I
PER-FRAME EXECUTION TIME

Resolution (px.)	Mean value (sec.)	Median value (sec.)
128x128	0.077	0.073
256x256	0.15	0.147
512x512	0.545	0.538

The results of our background removal process are depicted in Fig. 10. It can be observed that the output images have got lower resolution than the original images. This is because the model was trained on 128x128 images, and thus the resolution of the images suffered significantly.

Fig. 11 represents the results of the background replacement, RGB glitch and the person outline effects. Persons are segmented out from the images, and then the background is replaced with a given image. One major drawback is that the color grading of the person and the background do not match perfectly. Thus another effect could have been added to the final result to merge the colors of the background and the person so that it looks less artificial. Overall if the portrait segmentation is improved, the results of this effect would improve significantly.

VII. CONCLUSION AND FUTURE WORK

In the framework of this project, photo/video effects have been applied to portrait imagery/video based on semantic segmentation. The way person segmentation was done was by using convolutional neural networks. After the training, we separated the person from the background and applied the background replacement, RGB glitch, and person outline effects on the person segment.

To improve the results further, we recommend running the model training on at least 512x512 resolution images to have a higher resolution output. In order to produce better visual results further training should be performed on images that include full-body shots from a distance and multiple people.

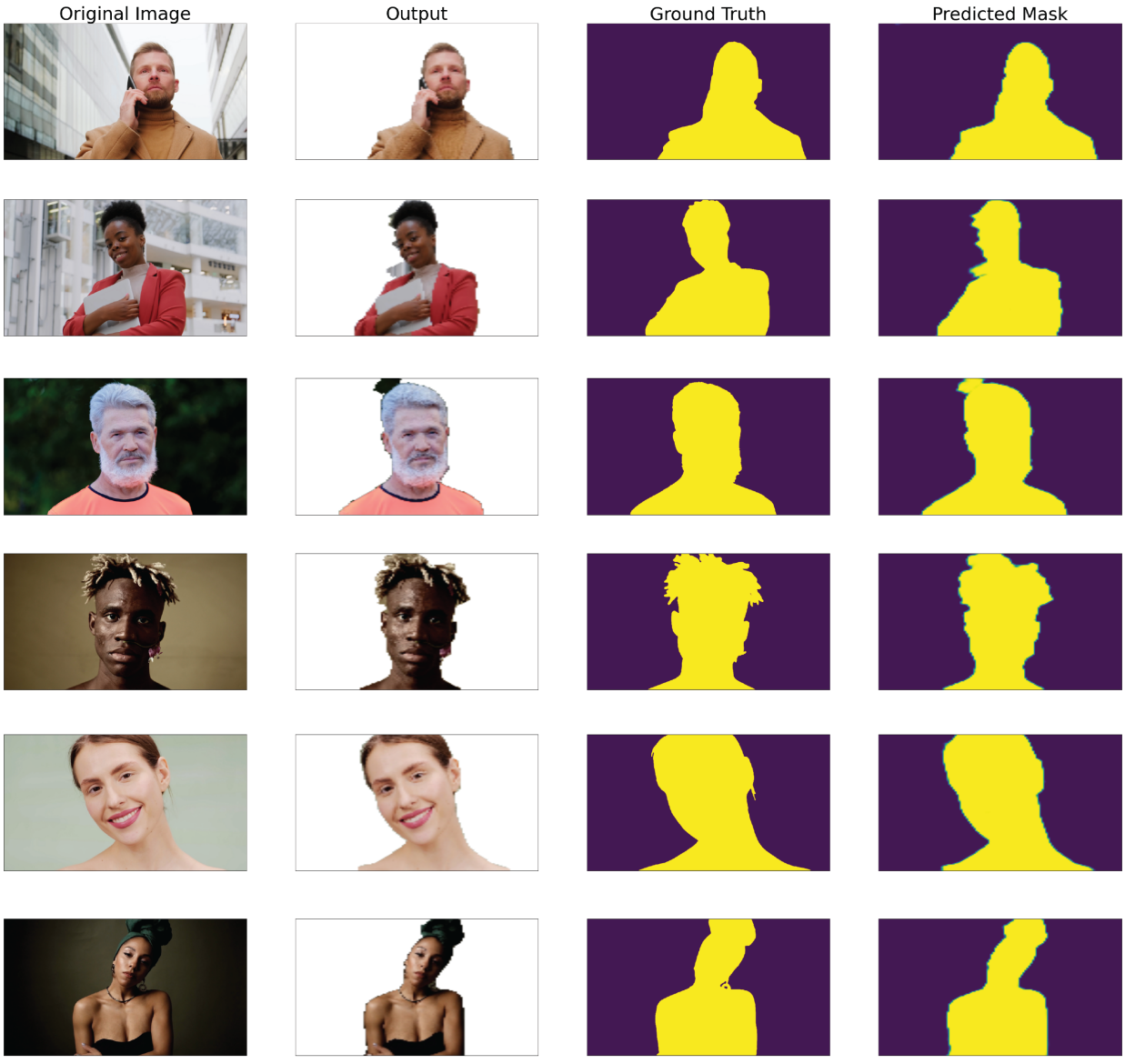

Fig. 10. Background removal results: Original images in the first column, the segmented person in the second column, ground truth mask in the third column, and the predicted mask by our model in the fourth column.

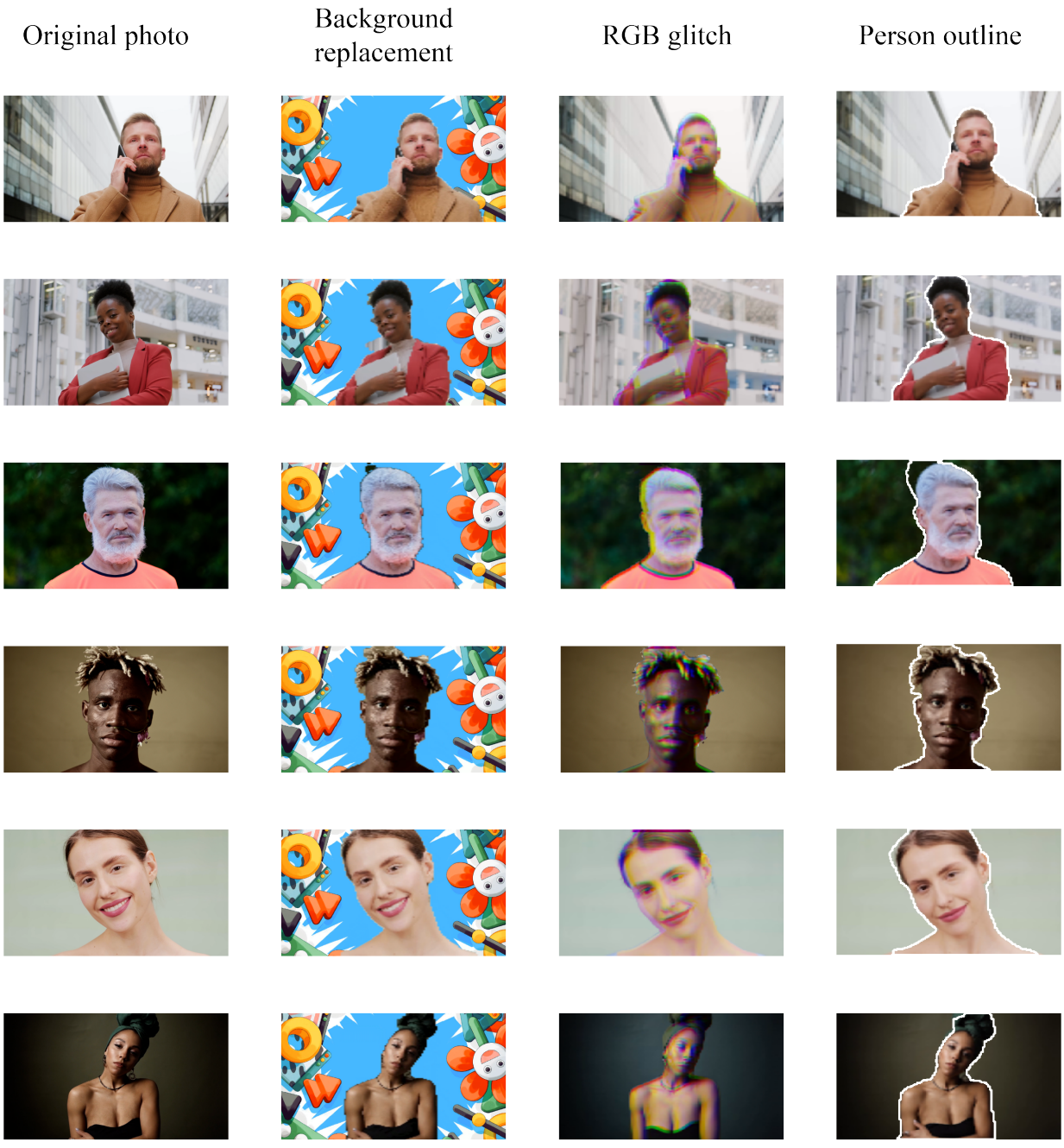

Fig. 11. Photo/Video effects results: Original images in the first column, background replacement effect in the second, RGB glitch in the third, and person outline in the fourth column.

REFERENCES

Al-amri, S. S., Kalyankar, N. V., & Khamitkar, S. D. (2010). Image segmentation by using thershod techniques. *JOURNAL OF COMPUTING*, *2*. https://arxiv.org/pdf/1005.4020.pdf

AppleInc. (2019). Advances in camera capture & photo segmentation - wwdc19 - videos. *Apple Developer*. Retrieved May 7, 2022, from https://developer.apple.com/videos/play/wwdc2019/225?time=2714

Barla, N. (2022). Panoptic segmentation: Definition, datasets & tutorial [2022]. *www.v7labs.com*. Retrieved May 16, 2022, from https://www.v7labs.com/blog/panoptic-segmentation-guide

Battini, D. (2018). Adam optimization algorithms in deep learning. *Tech-Quantum*. Retrieved May 15, 2022, from https://bit.ly/3LAFAsp

Duque-Arias, D., Velasco-Forero, S., Deschaud, J.-E., Goulette, F., Serna, A., Decencière, E., & Marcotegui, B. (2021). On power jaccard losses for semantic segmentation. *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, *5*. https://doi.org/10.5220/0010304005610568

Gupta, A., Harrison, P. J., Wieslander, H., Pielawski, N., Kartasalo, K., Partel, G., Solorzano, L., Suveer, A., Klemm, A. H., Spjuth, O., Sintorn, I.-M., & Wählby, C. (2018). Deep learning in image cytometry: A review. *Cytometry Part A*, *95*, 366–380. https://doi.org/10.1002/cyto.a.23701

Kaushik, R., & Kumar, S. (2019). Image segmentation using convolutional neural network. *International Journal of Scientific and Technology Research*, *8*. http://www.ijstr.org/final-print/nov2019/Image-Segmentation-Using-Convolutional-Neural-Network.pdf

Keymakr. (2021). Instance vs. semantic segmentation: What are the key differences? *Keymakr's Blog features the latest newsupdates*. https://keymakr.com/blog/instance-vs-semantic-segmentation/

Kirillov, A., He, K., Girshick, R., Rother, C., & Dollár, P. (2019). Panoptic segmentation. https://arxiv.org/pdf/1801.00868.pdf

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. https://arxiv.org/pdf/1411.4038.pdf

Prasad, S. (2020). What is image segmentation or segmentation in image processing? *Blogs & Updates on Data Science, Business Analytics, AI Machine Learning*. https://www.analytixlabs.co.in/blog/what-is-image-segmentation/

Ramachandran, P., Zoph, B., & Le Google Brain, Q. (2017). Searching for activation functions. https://arxiv.org/pdf/1710.05941.pdf

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *arXiv.org*. https://arxiv.org/abs/1505.04597

Shapiro, L. G., & Stockman, G. C. (2000). *Computer vision: Image segmentation*. Upper Saddle River Prentice Hall. http://nana.lecturer.pens.ac.id/index_files/referensi/computer_vision/Computer%20Vision.pdf

Sharif, M. S., Abbod, M., Amira, A., & Zaidi, H. (2010). Artificial neural network-based system for pet volume segmentation. *International Journal of Biomedical Imaging*, *2010*, 1–11. https://doi.org/10.1155/2010/105610

Sharma, M. (2020). Cluster-based image segmentation - python. *Medium*. https://towardsdatascience.com/cluster-based-image-segmentation-python-80a295f4f3a2

Vadapalli, P. (2021). Image segmentation techniques [step by step implementation]. *upGrad blog*. https://www.upgrad.com/blog/image-segmentation-techniques/

Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: An overview and application in radiology. *Insights into Imaging*, *9*, 611–629. https://doi.org/10.1007/s13244-018-0639-9

Yuheng, S., & Hao, Y. (2017). Image segmentation algorithms overview. https://arxiv.org/pdf/1707.02051.pdf

Zhang, J. (2019). Unet line by line explanation. *Medium*. https://towardsdatascience.com/unet-line-by-line-explanation-9b191c76baf5