

---

# Stylizing and Blending Portrait Picture into Painting Using Image Segmentation and Style Transfer

---

Lu Chen, Xun Zhou, Zirui Zheng Department of Computer Science  
Carnegie Mellon University Pittsburgh, PA 15213

## 1 Introduction

The field of computer vision is shifting from statistical methods to deep learning neural network methods. Deep learning methods are achieving state-of-the-art results on some specific problems, which includes image classification, object detection, object segmentation, image style transfer, image reconstruction and so on.

**Motivation** To explore practical value of such existing algorithms, we consider making a application based on combinations of them. In this paper, we develop an application to integrate portrait into paintings smartly. Specifically speaking, we make use of convolutional neural networks to do image segmentation and style transfer. As users choose a portrait and any style, they can create their own new pictures.

**Process** Our application has three steps. First, we do portrait segmentation by FCN; Second, we transfer style of portrait by Real-Time Style Transfer and Super-Resolution; Lastly, we resize the style-transferred image and blend it into painting.

For image segmentation, we use a fully automatic portrait segmentation method. It's based on contemporary classification networks. Such FCN trained end-to-end, pixels-to-pixels on semantic segmentation exceeds the state-of-the-art without further machinery. We also extend FCN by combining semantic information from shallower layer with that from deep layer, which enable convolutional network to do better segmentation masks.[7]

For style transfer, one approach is to train a feed-forward convolutional neural network in a supervised manner, using a per-pixel loss function to measure the difference between output and ground-truth images. However, the per-pixel losses used by these methods do not capture perceptual differences between output and ground-truth images. The other approach is to high-quality images can be generated using perceptual loss functions based not on differences between pixels but instead on differences between high-level image feature representations extracted from pre-trained convolutional neural networks. In this application, we combine the benefits of these two approaches. We train feed-forward transformation networks for image transformation tasks, and then, we train our networks using perceptual loss functions that depend on high-level features from a pre-trained loss network. This approach has been used for example by Jsutin et al for perceptual losses for real-time style transfer and super-resolution[6].

For portrait segmentation, we trained model on our own. For style transfer, we used two pre-trained model and trained four models by ourselves. Our results shows that all models converge in 20 epochs.

**Data** For portrait segmentaion, we used 1600 portrait sets of pictures as training data which each contains foreground and background. For Style transfer, we used PiperFrontal Faces dataset as training data which contains 60000 instances.

**Application** Our application can be widely used in lots of fields, such as making special effects for movies, editing pictures efficiently by adding some objects into them automatically, merging real person into classical drawings properly and so on.

## 2 Background

In our midway report, we start by extracting the person figure out of a picture. To identify people faces from photos, we first considered identify different objects in a picture, and this led us to R-CNN, Regions with CNN features, introduced by Ross Girshick et al in 2014[2]. R-CNN provides a smooth and fast way of object detection from photos. More specifically, we implemented a Mask R-CNN.

Mask R-CNN adds a branch to identify segmentations on each Region of Interest, in a pixel-to-pixel manner. It uses selective search to obtain regions:

- Initialize by generating many sub-regions.
- Recursively find similar regions, and join them together.
- Produce the candidate regions with results from Step 2.

we perform a Mask R-CNN derived from a pre-trained model from keras. We use the COCO dataset. As it is only our baseline method, we did not go far beyond the model itself. As a goal of stylization, we picked a small dataset of 30 images that could be potential stylization backgrounds to be our test data for model evaluation.

With help of AWS and GPU, we tune parameters like learning rate, mask shape, mask pool size, etc., and trained a model. It converges fast within 10 epochs.

As a result, our model can predict the pictures with relatively high accuracy.



Figure 1: Sample segmentation output from the Mask R-CNN architecture

However, after re-evaluating the outcome of the Mask R-CNN model and comparing it with other state-of-the-art architectures, we have identified the following drawbacks of the R-CNN architecture for the purpose of our task:

1. The most noticeable drawback of R-CNN architecture is that training is very expensive in both space and time.
2. R-CNN is more powerful for instance segmentation, while FCN is more powerful for semantic segmentation.

For our portrait segmentation task, under the assumption that the portrait picture contains only a single person, we do not need to identify multiple distinct person instances. Instead, we only need to classify each pixel into foreground (the people class) or the background. Therefore, we decided to proceed with the FCN architecture.

### 3 Related Work

#### 3.1 Image Segmentation

Shelhamer et al[7] introduced the fully convolutional networks (FCN) architecture based on contemporary classification networks (AlexNet, the VGG net, and GoogLeNet), and then fine-tuned the model for semantic segmentation tasks.

One important extension of FCN from previous network models is the skip architecture, which combines the semantic information from a deep layer with high-level information from a shallow layer with low-level information to produce accurate and detailed segmentations. The intuition is that the shallower layers are spatially dependent, which provides the *"where"* information, while the deeper layers has been convolved to extract global semantic information, which provides the *"what"* information.

Because of the pooling layer in the convolutional networks, the model is invariant to local translations. Thus, it makes convolutional networks especially powerful for segmentation tasks.

#### 3.2 Style Transfer

Johnson et al[9] introduced this Fast Style Network for style transformation training with a base architecture of the residual convolutional neural network, a training target generated from VGG16 model, and an environment of Chainer during implementation.

Nowadays we saw a lot of feed-forward deep convolutional neural networks on image deep learning subjects, such as FCN on semantic segmentation by Long et al[7] and implementation of CRF-RNN by Zheng et al[10]. There are also various models based on Residual CNN like Xie et al's work on image classification[11]. Here, we train a feed-forward network which can approximate optimality faster than models forward-backward propagation in each training step[9].

In addition, many of the related papers train models using per-pixel loss, while our model here directly optimizes the feature reconstruction loss using perceptual optimization[9]. This idea was inspired by Mahendran and Vedaldi's paper "Understanding Deep Image Representations by Inverting Them", and it can help better extract the image information from each layer of the network[12].

## 4 Methods

#### 4.1 Image Segmentation

**Architecture** The overall architecture is shown in the figure 4.

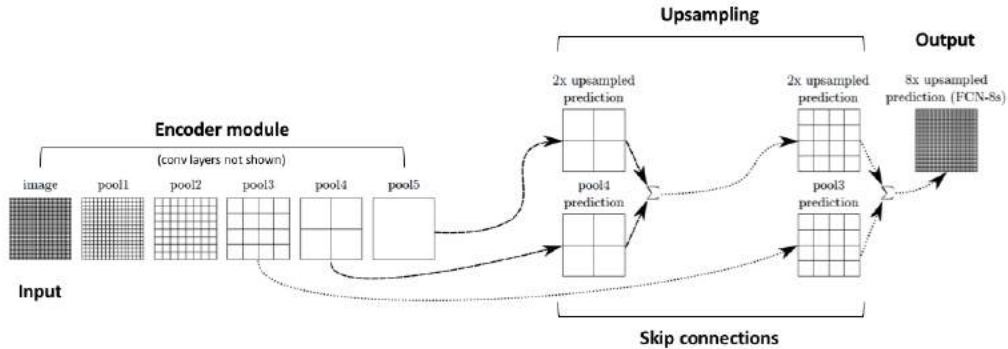


Figure 2: Architecture of FCN-8s (Modified from the figure in [7])

We explain the type and functionality of the core layers in FCN-8s as follows,

- **Convolution layer:** The convolution layer is the core building block of the network. Given an input matrix  $X \in \mathbb{R}^{H \times W \times C}$ , a kernel size  $k$ , number of kernels  $D$  and stride  $S$ , the convolution operation is as follows:

$$(X * K)_{ij} = \sum_{m=0}^{k-1} \sum_{n=0}^{k-1} \sum_{c=1}^C K_{m,n,c} \cdot X_{q,r,c} + b$$

where  $q = S * i + m$  and  $r = S * j + n$ . The convolution layer is able to extract relevant features from the images, for example, edges and corners.

- **Upsampling layer:** The upsampling layer, also known as deconvolution layer is used to connect coarse outputs to dense pixels in order to make the output size match the input size after downsampling in previous layers.
- **Pooling layer:** The pooling layer in our model uses max pooling. It helps reduce the spatial variance in the input image. Given a  $p \times p$  kernel, the max pooling operation is as follows:

$$\text{MAX\_POOL}(X)_{ij} = \max(X_{pi+r,pj+c})$$

where  $0 \leq r \leq p - 1$ , and  $0 \leq c \leq p - 1$

**Training Data** For the purpose of portrait segmentation, the training data we used are exclusively portrait pictures, 1600 in total. Each portrait is associated with a segmentation mask, indicating the foreground and the background.



Figure 3: Sample data from the training set with associated segmentation mask

## 4.2 Style Transfer

**Architecture** In this second part, we do the following: train a style-transfer model with a style image, and we use this model to generate a new picture with that specific style on our input image. This process was inspired by the idea of the “Perceptual Losses for Real-Time Style Transfer and Super-Resolution” by Johnson et al[9]. The whole process can be explained in Fig 1.

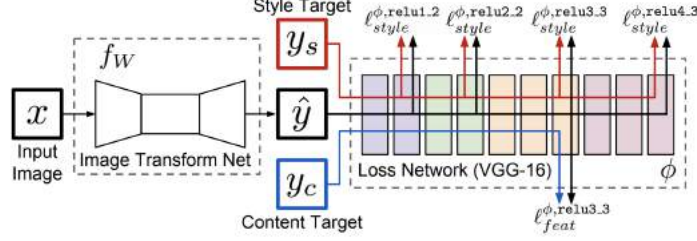


Figure 4: The model consists of two parts. Image Transform Net and Loss Network[9].

**Training Data** Unlike Johnson et al[9], who used COCO dataset in their paper[9], we used Piper Frontal Faces dataset as our training dataset for the style transformation model. COCO dataset contains all kinds of pictures in life, while Piper dataset contains 60000 instances of 2000 individuals collected from public Flickr photo albums[17].

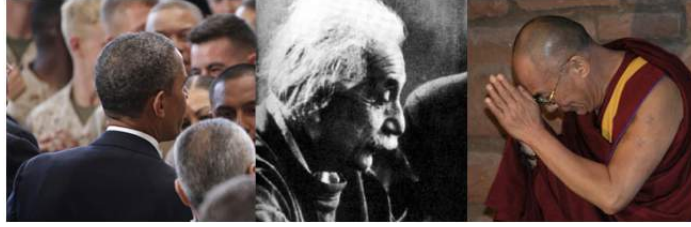


Figure 5: Sample pictures in Piper dataset.

**Our Style Transfer process consists of three main steps:**

**(i) Style construction.** We first train a model with a given style. VGG16 model here is used as our ‘label’. The VGG16 model has an excellent performance in image localization and the pre-trained caffe model is available online. It works well extracting the key information of the image. By passing our style image into VGG16 model and generating the target, we transform our style image with a stack of small CNN filters: four 2x2 and nine 3x3, as VGG16 showed that the depth of network can largely improve the performance[14]. By evaluating the perceptual loss between training dataset and the target, we build one model, which stores over 1 million variables, for one style. Suggested by Johnson et al[9], we trained the model with 2 epochs on over 34000 people images.

**(ii) Content reconstruction.** Secondly, we take the output from our first model (Profile Detection) and reconstruct a new image by applying the model trained above. Usually this process won’t take long. From the results later Table 1, we will see that the high-level content of the image is well-preserved, which is consistent with the results by Gayts et al[15]. It can take any-size pictures but before we generate our final input, we will modify the size as a way to communicate with both models.

**(iii) Style mixture.** Lastly, we need to blend the cropped-out profile picture into the style image. Right now, we offer the user the option to choose the location for blend-in, with terminal operations ‘-x’ and ‘-y’. In future, we aim to study some deep learning automatic blend-in techniques, such as those in “Learning to Blend Photos” by Hung et al[16].

### 4.3 Integration

The original portrait,  $P \in \mathbb{R}^{\{H,W,3\}}$  is first passed as an input into the FCN-8s model fine-tuned on the portrait data set. The output is a binary mask matrix  $M \in \{0,1\}^{H,W}$ . We then passed the original portrait  $P$  into the style transfer model, and get transferred image  $M' \in \mathbb{R}^{\{H,W,3\}}$  as the

output. We applied the mask  $M$  onto the transferred image  $P'$  with the formula,

$$\text{Mask}(P', M)_{ij} = \begin{cases} (P'_{ij1}, P'_{ij2}, P'_{ij3}, 0) & \text{if } M_{ij} = 0 \\ (P'_{ij1}, P'_{ij2}, P'_{ij3}, 255) & \text{if } M_{ij} = 1 \end{cases}$$

where the result image  $P'' \in \mathbb{R}^{H,W,4}$  is of type RGBA. The final step is to resize  $P''$  to the height and width specified by the user, and merge  $P''$  with the given background.

We also benchmark the integration process on a 2.6 GHz Intel Core i7 CPU. Given an input image of size  $600 \times 800 \times 3$ , the portrait segmentation phase takes 7.56 seconds, and the style transfer phase takes 6.75 seconds. But since these two phases do not depend on each other, we can parallelize the integration process and achieve further improvement.

## 5 Results

### 5.1 Quantitive Results

**Image Segmentation** We started with the pre-trained FCN-8s model (PASCAL VOC models). We first changed the number of outputs to two in the deconvolution layers, because we only need to classify each pixel to two classes, foreground or background. We initialize the unknown weights in the deconvolution layers with random values and fine-tune on our portrait data set with a learning rate of  $10^{-4}$ . The model is trained on AWS with a single Nvidia Tesla K80 GPU for 21 epochs (16800 SGD steps). The model converged after 20 epochs, and the cross-entropy loss we got is 0.17. The training loss is shown in figure 6

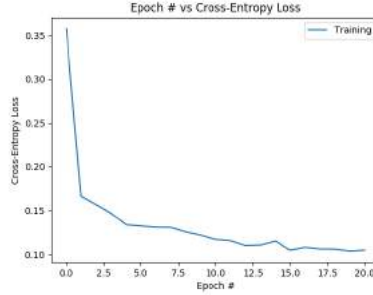


Figure 6: Training Loss over Epoch Number

**Style Transfer** We recorded the perceptual losses during training, and they have similar accuracy and trends comparing with Johnson et al.

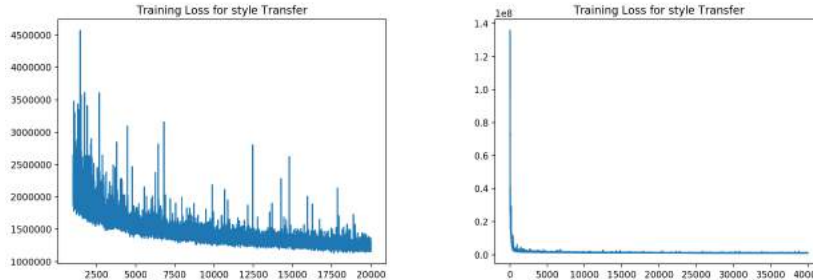


Figure 7: Training losses within first 20000 training and first 40000 trainings.



## 5.2 Result Examples





































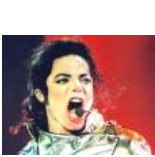





Image Segmentation		Style Transfer		Blend	
Input	Segmentation	Style	Portrait Style Transfer	Resize	Result
					
					
					
					
					
					
					

Table 1: A few examples of our experiments. Three steps are shown: image segmentation, style transfer and integration.

**Comparison with prior work** Compared to our midway report work, we've achieved a great progress. In terms of image segmentation, we used a more efficient network to get portrait. Also, we successfully did style transfer to the portrait and integrate with the painting, which leads to a good effect.

## 6 Discussion and Analysis

In our application, we did a we did a high performance automatic method to integrate person into painting. The system is built on FCN to do image segmentation and CNN to do style transfer. For image segmentation, we construct a large portrait image dataset with enough portrait segmentation and groundtruth data to enable effective training and testing of our model. For style transfer, we trained our model based on different painting styles.

After a thorough literature review of related works, we proposed some potential bases for our future work.

To further improve the current model, we have to pay attention to the mixed pixels, the pixels that belong both to the foreground and the background. Therefore, we are interested in looking at the image matting technique, which could help identify the  $\alpha$  values for the mixed pixels, so that the separated foreground has smoother edges.

The matting problem is ill-posed and severely under-constrained. The equation (1) given above has 7 unknowns ( $F_i, B_i$  for each of the 3 color channel and  $\alpha_i$ ), and 3 equations. The severe under-constrained problem are often solved using *trimap*, i.e. labeling some pixels that completely belongs to foreground or background. Usually, the user segments the input image into three regions: definite foreground, definite background, and unknown region, i.e. trimap. The problem can then be reduced, because it can learn the values of  $F_i, B_i$  and  $\alpha_i$  for the pixels in the unknown region based on the given information of definite foreground and definite background. However, providing an accurate user input requires a lot of human effort, and can be difficult for non-professional users without sufficient matting knowledge. Some recent works have shown the possibilities of automatic portrait matting, which relieve the burden for users. Our future work will experiment on applying automatic matting to our task, and compare the result with the existing method using trimap. Our current plan is to train a fully convolutional neural network (FCNN) on portrait datasets.

There are also some aspects we need to improve in future work. In terms of image segmentation, our application could fail when the background and foreground have very small contrast. Also, our method only focuses on portrait of human beings. We treat this as the limitation of our method. In the future, we will improve our model for higher accuracy and extend the framework to image segmentation for different objects. Additionally, we plan to make the last step (integration) automatic by building a new CNN model rather than resize and blend manually.



## 7 References

- [1] Girshick, R., Donahue, J., Darrell, T., Malik, J. (2014). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. 2014 IEEE Conference on Computer Vision and Pattern Recognition. doi:10.1109/cvpr.2014.81
- [2] Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick. Mask R-CNN
- [3] Gandhi, R. (2018, July 09). R-CNN, Fast R-CNN, Faster R-CNN, YOLO - Object Detection Algorithms. Retrieved from <https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e>
- [4] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge. Image Style Transfer Using Convolutional Neural Networks
- [5] Xiaoyong Shen-Aaron Hertzmann-Jiaya Jia-Sylvain Paris-Brian Price-Eli Shechtman-Ian Sachs - Computer Graphics Forum - 2016
- [6] Perceptual Losses For Real-time Style Transfer and Super-resolution Justin Johnson-Alexandre Alahi-Li Fei-Fei - Computer Vision – Eccv 2016 Lecture Notes in Computer Science - 2016
- [7] Long, J., Shelhamer, E., Darrell, T. (n.d.). Fully Convolutional Networks for Semantic Segmentation. Retrieved from [https://people.eecs.berkeley.edu/~jonlong/long\\_shelhamer\\_fcn.pdf](https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf)
- [8] Radford, Alec, Metz, Luke, Soumith. (2016, January 07). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. Retrieved from <https://arxiv.org/abs/1511.06434>
- [9] Johnson, J., Alahi, A., Fei-Fei, L. (2016). Perceptual Losses for Real-Time Style Transfer and Super-Resolution. Computer Vision – ECCV 2016 Lecture Notes in Computer Science, 694-711. doi:10.1007/978-3-319-46475-6\_43
- [10] Zheng, S., Jayasumana, S., Romera-Paredes, B. (n.d.). Conditional Random Fields as Recurrent Neural Networks. Retrieved from <https://arxiv.org/pdf/1502.03240.pdf>
- [11] Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K. (2017). Aggregated Residual Transformations for Deep Neural Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2017.634
- [12] Mahendran, A., Vedaldi, A. (2015). Understanding deep image representations by inverting them. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2015.7299155
- [13] Simonyan, Karen, Zisserman, Andrew. (2015, April 10). Very Deep Convolutional Networks for Large-Scale Image Recognition. Retrieved from <https://arxiv.org/abs/1409.1556>
- [14] VGG16 - Convolutional Network for Classification and Detection. (2018, November 21). Retrieved from <https://neurohive.io/en/popular-networks/vgg16/>
- [15] A., L., Ecker, S., A., Bethge, Matthias. (2015, September 02). A Neural Algorithm of Artistic Style. Retrieved from <https://arxiv.org/abs/1508.06576>
- [16] Hung, W., Zhang, J., Shen, X. (n.d.). Learning to Blend Photos - [faculty.ucmerced.edu](http://faculty.ucmerced.edu/mhyang/papers/eccv2018_blending.pdf). Retrieved from [http://faculty.ucmerced.edu/mhyang/papers/eccv2018\\_blending.pdf](http://faculty.ucmerced.edu/mhyang/papers/eccv2018_blending.pdf)
- [17] Zhang, N., Paluri, M., Taigman, Y., Fergus, R., Bourdev, L. (2015). Beyond frontal faces: Improving Person Recognition using multiple cues. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2015.7299113