ARTISTIC STYLE TRANSFER - AN EXPLORATION

Sidharrth Nagappan *Nadia Ahmad Pirdaus* *Cornelius Pang*

Faculty of Computing & Informatics, Multimedia University, Cyberjaya

ABSTRACT

This work proposes a web application that demonstrates the power of neural style transfer when coupled with foreground-background and automated semantic segmentation. Pioneered training on the Stanford Background dataset, the style transfer app is capable of generating remarkably aesthetic results albeit unconventional ones at times. The main motivation of this application is to enable users to convert their memorable moments captured in images into wonderful works of art from various well-known styles. Reference code for this project is available at: https://github.com/sidharrth2002/neural-style-transfer

Index Terms — Style Transfer, Computer Vision, Deep Learning, Image Processing, Segmentation Masks

1. INTRODUCTION

Pablo Picasso once said "good artists copy, great artists steal". In art, there have been countless attempts shown by artists to compose unique visual experiences by combining the style of an image and the content of another image. Recent advances in Convolutional Neural Networks have enabled researchers to create artificial systems that generate artistic images with high perceptual accuracy. The problem that we propose to study is style transfer - a process of transferring the semantic content of an image to other images of different styles.

Style transfer algorithms have shown remarkable performance in recent years. In this work, we propose a semantically segmented style transfer algorithm that combines the benefits of scene parsing with style transfer to generate aesthetically interesting graphics. A combination of several style transfer algorithms, segmentation models and image processing techniques will be fused into a single pipeline for what we coin as *semantically segmented artistic style transfer.*

2. BACKGROUND STUDIES & SIMILAR APPLICATIONS

The idea of artistic style transfer firstly originated from non-photorealistic rendering (Kyprianidis et al., 2012), and is closely related to the problem of texture synthesis (Efros & Leung, 2001). Older techniques as such mostly relied on the statistics of pixel values in images while disregarding semantic structure. Recent developments in style transfer uses learning-based approaches such as deep learning architectures, as it is shown to be capable of synthesizing new high perceptual quality images with remarkable results.

In a recent work done by Gatys et al. (2015), the authors demonstrated the capabilities of Convolutional Neural Networks for artistic style transfer using a 19-layer VGG network, which sparked an interest in the research community towards studying the performance of various deep learning architectures. The authors' proposed deep architecture was capable of separating the style and content representations within images.

With the recent explosion of visual data on the web for the past few years, real world implementations of computer vision algorithms such as artistic portrait filters in social media applications and photo editing applications have also contributed to the growing interest in studying style transfer.

To assist in the app development process, several similar applications were tried out to gain insights on currently available features and its userbase's expectations. After some testing, we discovered that certain apps like 'Real Photo Art Painting' struggle in styling faces, and tend to only work well on the image's background. This shows that some apps opt for a more pixel-based approach when carrying out style transfer, as opposed to a more subject-based approach. There is also a lack of customization offered, as users are only able to apply a style of choice to the *whole* image. So if for instance they wish to style only the background but not themselves, an app akin to 'Real Photo Art Painting' is not equipped with functionalities to do so.

Figure 2.1: Input image (left) and how it looks like post-styling (right)

While the above case is true for most generic apps that can be downloaded from Google's Play Store, higher tier ones like 'Lensa' have begun to provide more freedom of customization to their user base by allowing them to style the background and foreground of an image separately. A foreground-background segmentation model behind the works allows the user to customize an image based on the segmented masks generated by that model.

Even 'Photoshop' is equipped with a fairly comprehensive style transfer feature. One downside that all three of these apps share is that users are restricted to only the predefined styles. There is no option to bring in and interpret our own personalized style yet, which is a future work that could be expanded upon.

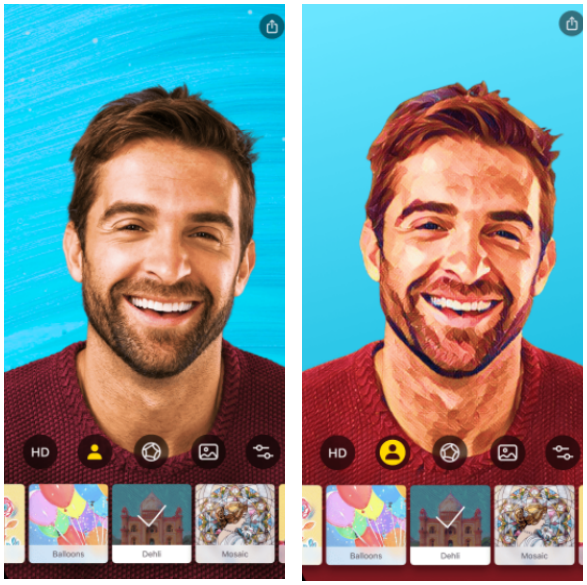

Figure 2.2: Prisma Lab's (2016) Lensa allows users to style an image's background (left) or main content (right) via foreground-background segmentation

In line with what we have found thus far, many applications only carry out style transfer on the entirety of an image, without providing users with the freedom to choose which parts to style. Lensa is a step-up from this as it allows users to apply different styles to the background and foreground of an image. Extending from this, we propose a style transfer approach that combines semantic segmentation via UNet to partition the image into parts of interest with Magenta as the fast style transfer model. We also include an option for foreground-background segmentation using Detectron2 and GrabCut for those who prefer a more typical way of styling an image.

3. DATASETS

Segmentation

The project will employ a Stanford Background dataset that features images taken in different outdoor scenes with the occasional cameo from passersby. It contains RGB colour code mappings for different classes in labels, with 9 labels altogether:

- sky
- tree
- road
- grass
- water
- building
- mountain
- foreground
- unknown

As observed, the majority of labels are what we would term "style transferrable", such that it is meaningful to perceive them in different styles. This is in comparison to the vast number of labels in the COCO dataset, such as "water dispenser". Therefore, the Stanford Background dataset will be used to assess the quality of outputs on the app.

Purpose for Choosing This Dataset

There are several reasons why this dataset was chosen to provide training images for our model. For one, the dataset sports a total of 715 images while being merely 14 MB in size, making it particularly lightweight. This will allow space to be conserved, and reduce computation time needed to train the model as there are not that many images to begin with. Furthermore, the dataset aligns with the concept and idea of style transfer, where the goal is not to style everything, but rather only certain objects of note. The minimalist nature of the class labels in this dataset support this idea, making it easier for us to manipulate them accordingly.

Figure 3.1: Sample of images found in the Stanford Background dataset

The images uploaded by the user onto the app for styling will form the test dataset for this project.

Style Transfer

The style transfer network will use the following style images. Since we are not training the style transfer network from scratch, six images are sufficient:

1. Kandinsky
2. Shipwreck
3. Seated Nude
4. Woman with a Hat
5. Starry Night
6. The Scream

Kandinsky

Shipwreck

Seated Nude

Woman with a Hat

Starry Night

The Scream

Figure 3.2: Chosen paintings for style transfer

Such are iconic paintings, which are inherently a part of Magenta (style transfer model)'s pretraining. They have shown remarkable results for single style transfer, but their performance for multiple semantic style transfer remains untested.

4. PROPOSED DESIGN AND IMPLEMENTATION

There are four distinctive parts of this project, namely the finetuning of a segmentation model, selection of a fast style transfer model, the merging of semantic segmentation with style transfer and model deployment. The segmentation models will be trained on Pytorch, while the fast style transfer model will use Tensorflow.

Semantic Segmentation using UNet

A UNet segmentation model consists of two components: an encoder and a decoder. Given data limitations, we choose to use a pretrained ResNet model as the encoder using ImageNet weights. A separate decoder is attached and fine tuned to the Stanford Backgrounds dataset.

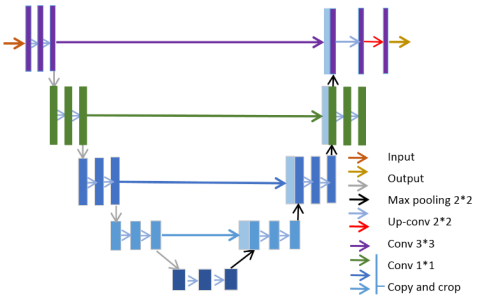

Figure 4.1: Encoder-decoder architecture of UNet

Stanford Backgrounds only consists of 800 images, which is insufficient for training a decoder from scratch. Therefore, augmentation strategies consist of random horizontal and vertical flipping, rotations and distortions. The Pytorch preprocessing pipeline also:

1. Accommodates the resizing of images into a (320, 320) shape for model input by applying bordering padding
2. Transposing the tensor to (channels, height, width), as opposed to Tensorflow's (height, width, channels)
3. One-hot encoding the segment labels for each pixel

Training is done for a total of 60 epochs across two training runs (due to Google Colab's computing limitations), with the segmentation model saved at the highest IOU score, which is 0.8589. An Adam optimiser with a learning rate of 0.0001 is used. Code for fine tuning UNet is inspired by the work of Balraj Ashwath.

Detectron2 x GrabCut

Detectron2 is a detection and segmentation algorithm based on Mask R-CNN. It is quite an extensive model, so we opted for its pretrained variant using Model Zoo weights. Alone, it can generate segmentation masks without the need to predefine a bounding box or rely on human supervision. However, as with other deep learning methods when it comes to segmentation, the masks generated are not perfect and tend to either cut off or oversegment a region of interest. Therefore, we decided to combine Detectron2 with GrabCut to further refine the edges and produce a cleaner mask.

The main issue with masks generated by Detectron2's algorithm is that some background pixels end up being classified as foreground pixels during the segmentation process, leading to oversengmentation. GrabCut serves to rectify this by taking said mask as an approximation of the region of interest and builds upon that by iteratively narrowing down pixels that represent the definite foreground and background of an image to improve its segmented output.

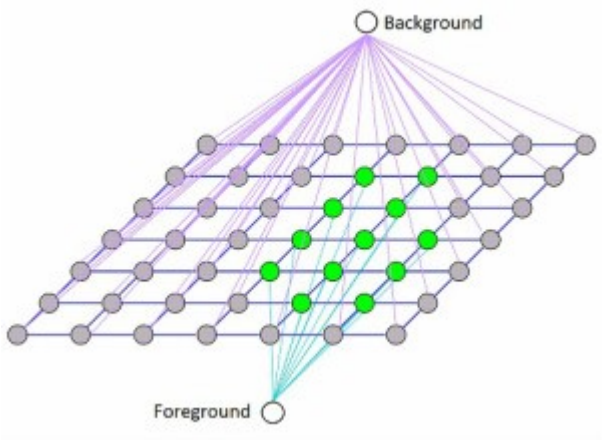

Figure 4.2: GrabCut classifies neighbouring pixels as foreground or background based on the similarity of their color distribution

Fast Style Transfer

The style transfer module of our pipeline is largely untouched. A pretrained fast style transfer model named Magenta is made available by Google through Tensorflow Hub. The functionality of the module is made available through an abstraction function, that inputs a 2-dimensional content image, style image (from a preset assortment) and outputs the styled version of the content image.

The original style transfer algorithm by Gatys et al. (2015) involved repeated forward and backpropagation through the network, the creation of Gram matrices and the calculation of mean squared error using multiple Gram matrices between the style image and the output image. The Gram matrix is a spatial representation of an image's features and the closer it is to the style image, the more effective the output is at emulating the style. While this is the preliminary approach, fast, real-time style transfer (Ghiasi, Lee, Kudlur, Dumoulin & Shlens, 2017) makes way for a more efficient representation. A single encoder-decoder architecture is trained to learn representations over a wide range of paintings, in an agnostic manner. When used for prediction, only the normalization parameters are optimised, via conditional instance normalization. This forms Magenta's backbone.

$L_c(x,\ c)$ is the content loss and $L_s(x,\ s)$ is the style loss. $G[f_i(x)]$ is the Gram matrix at a particular layer.

$$L_s(x,\ s) = \sum_{i=S} \frac{1}{n_i} || G[f_i(x)] - G[f_i(s)] ||_F^2$$

$$L_c(x,\ c) = \sum_{i=C} \frac{1}{n_j} || f_j(x) - f_j(c) ||_2^2$$

$$min\ L_c(x,\ c) + \lambda\ _sL\ _s(x,\ s)$$

Gram Matrices

Every convolutional layer learns a set of features. Feature maps are learnt for dots, patterns, textures and other such distinctive elements of images. The Gram Matrix essentially finds a correlation between these features by multiplying a feature matrix with its transposed form. It is now able to correlate certain patterns with others. This correlation quantifies what we refer to as the style of an image.

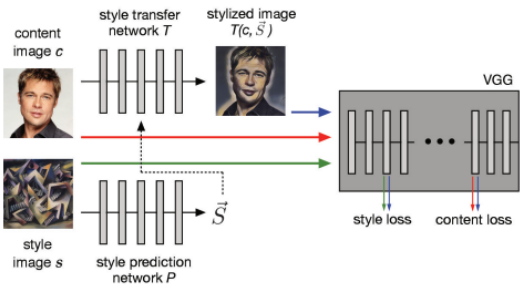

Figure 4.3: Flow of a style prediction network proposed by Ghiasi et al., (2017)

Figure 4.3 outlines an illustration of a content image, style image, style prediction network (learning instance normalization parameters from the style image to be used by style transfer network T) and a VGG-19 model that learns a style loss and a content loss that are jointly optimized.

Semantic Fast Style Transfer

This stage involves combining the segmentation map generated by UNet with the style transfer functionality of Magenta. To do this:

1. The UNet module label encodes each pixel of the image into one of the preset Stanford Background annotations.
2. Based on the segments visible in the image, a form is dynamically generated to allow the user to select a style for each segment.

Figure 4.4: Example of Segment Map

3. A dictionary is created mapping a segment label to a style.
4. The original content image, the label mask generated by UNet and the mapping is passed into a semantic style transfer function that creates separate layers. Each layer only sports the contents of one segment and all other elements are zeroed out. The layer is passed into the style transfer module.
5. Once multiple layers (segments) are styled, they are combined into a singular image. Each segment of the image is now styled differently, as per the user's specification.
6. If five segments are visible in the image, the style transfer module will be called five times before a pixel-wise amalgamation.

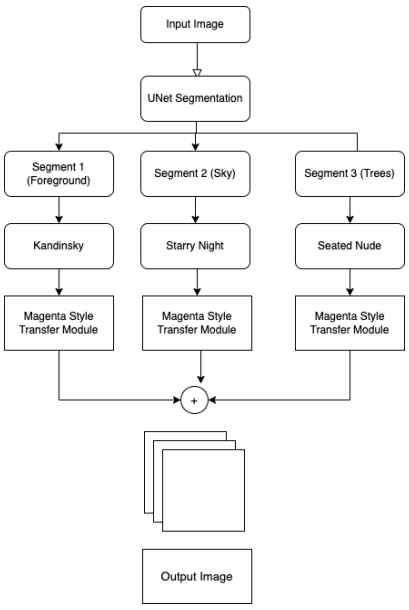

Figure 4.5: Generic pipeline showing the amalgamation of *multiple semantic style transfer*

We are not the first to evaluate masked style transfer, for Makow and Hernandez (2017) have experimented with the idea before us. They used the Microsoft COCO dataset using 5 styles. However, their pipeline does not involve automatic segmentation; one has to provide a mask separating elements of the image as an input to the module.

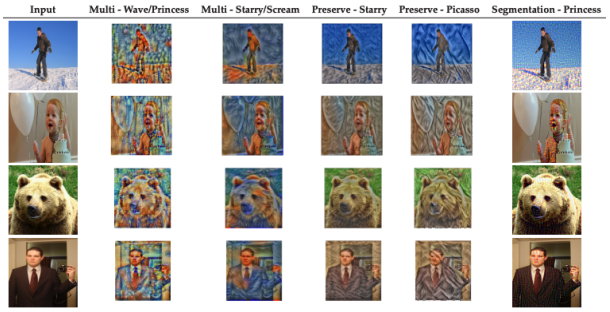

Figure 4.6: Mixed Style Transfer results of Makow and Hernandez (2017)

App Features

1. Single Style Transfer - Apply one style to the entire image (no tuning from our side, using only a pretrained model)

   A style is chosen from a preset set of styles before calling the Magenta module.

2. Custom foreground and background mask - Instead of automatic segmentation, allow users to specify semantic regions of interest for stylization. Allow users to manually mark the foreground and background before being prompted to select styles for each segment. This is an extension of the work of Makow and Hernandez (2017).

3. Foreground extraction using Detectron2 with GrabCut edge tuning

   Upload photo, call Detectron2 model for segmentation and tune edges before prompting style selection for individual segments. The code was inspired by Shubham Bindal's implementation of automated segmentation mask generation.

4. Semantic style transfer using UNet - Segment image into 8 possible semantic labels, user chooses style for each segment, each segment is styled independently before being combined into the final image.

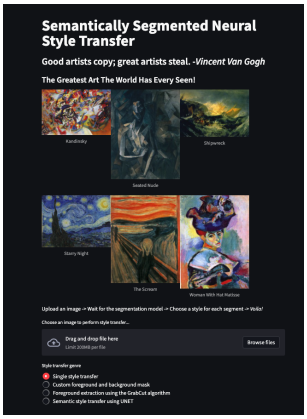

Figure 4.7: Screenshot of web app homepage, listing available features

5. RESULT EVALUATION

Our app is capable of applying a style to the whole image, or varying styles to its segmented parts of interest. As such, we will analyze the generated output for both of these cases as a form of qualitative analysis.

Styling a Whole Image

Figure 5.1: Styling an image in its whole

Figure 5.1 depicts the result of styling an image based on Matisse's Woman with a Hat. All parts of the image have been styled accordingly based on the style of choice, so we consider this feature to be working as planned.

Styling an Image Based on Its Foreground & Background

Figure 5.2: Styling only the background of an image

One of the features of our app is to allow the user to style either the foreground or background of an image, which will be automatically segmented beforehand with the help of a model based on Detectron2 and GrabCut. Figure 5.2 showcases an example of applying Shipwreck style only to an image's backdrop. Based on the output, the segmentation model was able to clearly partition the image into two major parts, with the girl representing the foreground and the rest as the background. The Shipwreck style also blends in well with the input image. The next figure is an example of styling only the foreground, and features a picture of Obama. The resulting segmentation mask deduced by the model is also included.

Figure 5.3: Styling only the foreground of an image

As we can see in Figure 5.3, the mask has relatively clean and smooth edges which can be attributed to the edge trimming done with GrabCut.

Freedom in Image Styling with Semantic Segmentation

Figure 5.4: Styling an image based on semantic segmentation

Furthermore, our app is able to semantically segment an image into regions of interest based on a set of generalized labels corresponding to the Stanford Background dataset.

Figure 5.4 gives a preview of the segmentation and style transfer models working in tandem to produce an image comprising various styles on the boy's face and shirt, as well as the background.

Figure 5.5: Mixing different styles into an image of Obama based on the generated mask

As per the above segmentation mask, it is clear that Obama could have been better extracted from the background of this image. Instead, the model seems to have mistaken the flag at the left side of the picture as a part of Obama himself. Compared to the segmentation mask in Figure 5.3, this one is not as clean. Although the styled variation does not look inherently bad, it may be better to go for foreground-background segmentation instead for a more ideal output.

Since the UNet segmentation model was trained on images that put more emphasis on backgrounds and scenery, it risks returning an iffy looking mask if the image is portrait-based like a selfie. An example of such a case can be observed in the following figure.

Figure 5.6: UNet did not segment the whole of Nicki Minaj, leading to an awkwardly stylized image

Figure 5.6 features a close-up of Nicki Minaj that was not accurately segmented, which best portrays how subjective the art of image styling can be. From a technical standpoint, the UNet model was not able to correctly segment out Nicki Minaj as a single entity from the image, unlike its background that is clearly segmented. Inspite of this, the image does not look horrendous and can even pass for a creative attempt at styling images. This leads us to consider this as a successful failure case, where the result is aesthetically pleasing but does not make sense logically.

Lastly, the app was tested on images that focus more on scenery rather than the people in it. Since this is what the UNet segmentation model was trained with, we expected it to return clear-cut results and it did not disappoint.

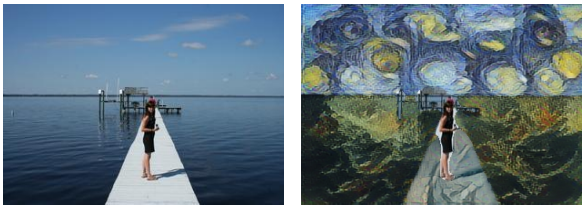

Figure 5.7: Applying different styles to the sky, sea and pathway

Figure 5.8: Applying different styles to the sky, sea and sand

Figures 5.7 and 5.8 highlight how well the UNet model performs on images that have the scenery as its focal point. This can be attributed to how the model's training data is made up of similarly structured images.

In short, while our app accepts any kind of image to be styled, certain images like selfies and close-ups of people may produce a dodgy looking output if the user opts for semantic segmentation via UNet first.

A workaround would be to go for single image style transfer or to segment only its foreground and background, as these approaches would yield more sensible looking results.

Figure 5.9: Masks generated for foreground-background segmentation (left) and semantic segmentation (right)

In line with our goal to measure the performance and reliability of the proposed style transfer app, a survey was carried out to gauge the overall user satisfaction of our application, as well as to gather useful feedback for potential future work. We propose to collect data from a user base of various age groups in order to better understand each generation's general usage of the application. Given time limitations, we only obtain 25 responses. This is sufficient as the questionnaire is quite straightforward.

Scales used in the problems for this survey encompass likert scales, short questions, and single-selections. The survey prompts the user to:

- Select their age group
- Select the frequency of which they use cartoonization, style transfer applications or Instagram filters
- Rate the aesthetic quality of several pictures that have been stylized by the app
- Provide any further suggestions

The results of the survey are as follows.

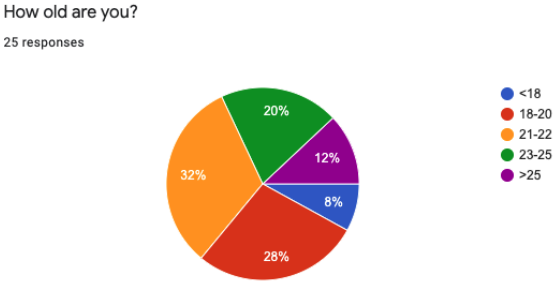

Figure 5.10: Age range of survey participants

Most users are in the university age range.

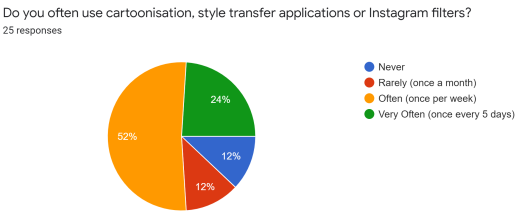

Figure 5.11: Frequency of which the participants use cartoonization, style transfer applications or Instagram filters

It is not surprising to see that most people use style transfer unintentionally on a daily basis through Instagram filters and other social media platforms.

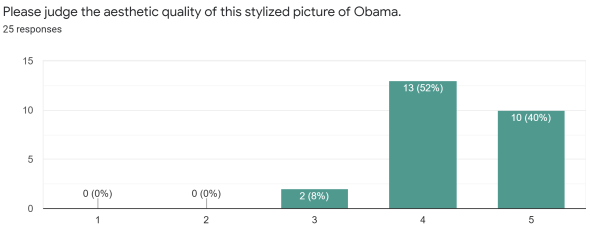

Figure 5.12: Overall aesthetic rating of a successful case of style transfer

The Obama picture was perfectly segmented and styled, so results are skewed towards the higher end.

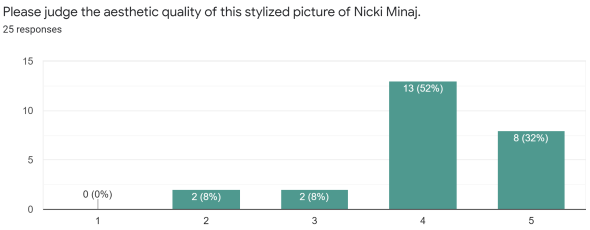

Figure 5.13: Overall aesthetic rating of a successful failure case of style transfer

As described above, the segmentation and stylization of Nicki Minaj's portrait is an example of a "successful failure". Some found it almost unintelligible, but others gave it a high aesthetic rating, further attesting to the subjectivity of image aesthetics.

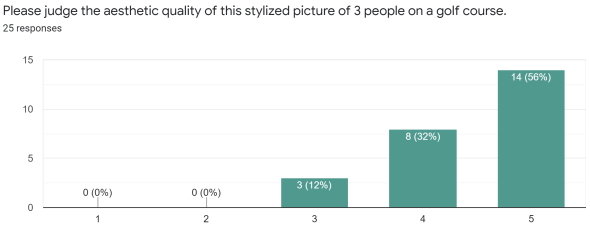

Figure 5.14: Overall aesthetic rating of another successful case of style transfer

This image did very well because it was lifted directly from the Stanford dataset. The segmentation was near perfect with a high IOU score and stylization was aesthetically pleasing.

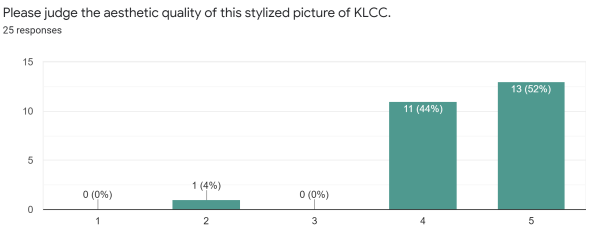

Figure 5.15: Overall aesthetic rating of a failure case of style transfer

UNet was not trained on such complex cityscapes, so the segmentation was rather abysmal, but the stylization produced an interesting mosaic. Although meaningless in terms of stylization, it looked rather interesting and the form responses seem to echo this hypothesis.

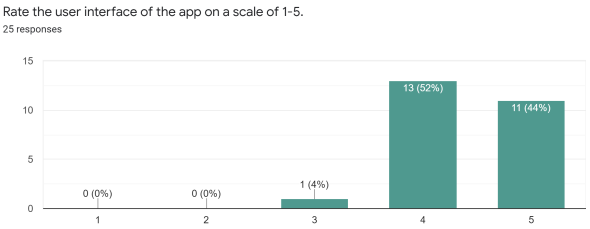

Figure 5.16: Rating of the app's user interface

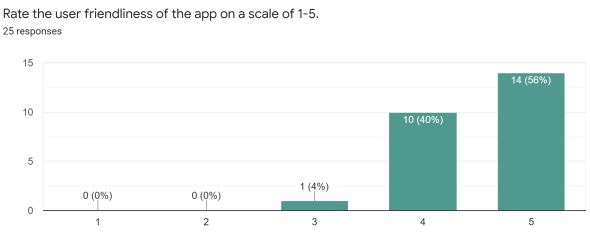

Figure 5.17: Rating of the app's user friendliness

Based on Figures 5.16 and 5.17, all 25 participants responded positively to both the app's user interface and user friendliness. The simplicity and straightforwardness of the user interface likely played a integral role in making the app seem user friendly, as it provides minimal room for confusion on what to do.

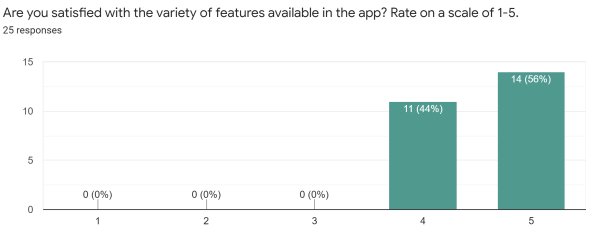

Figure 5.18: Satisfaction towards current features available in the app

Figure 5.18 conveys that the features made available in the present version of the app are enough to deliver an enjoyable experience to the user.

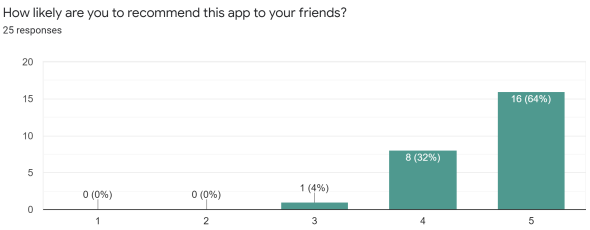

Figure 5.19: Likelihood of recommending the app to others

More than ⅗ of the participants claim that they would definitely recommend our app to others, signifying that the app has an audience and market value. With a few tweaks, this app has potential for deployment.

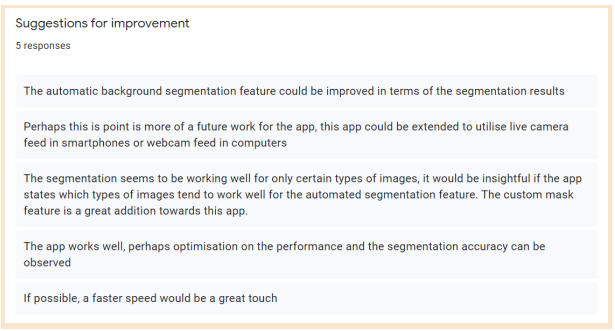

Figure 5.20: Sample of suggestions left by participants

Possible improvements include better tuning the segmentation algorithm and speeding up the processing. Since there are a variety of models and pipelines running simultaneously, optimising the app in terms of performance and speed should definitely be a priority.

6. LIMITATIONS AND FUTURE WORK

Potential for Deployment

As mentioned, the app has the following models running simultaneously to power key features:

1. UNet trained Pytorch model
2. Magenta style transfer trained Tensorflow Lite model (lightweight because of minification by Google)
3. Detectron2 segmentation
4. Grabcut (just one function, so unproblematic)

Regardless, all models can be minified for mobile deployment. The current UNet model stands at 120MB, but can be reduced to just 10MB using pytorch's mobile optimisation. Furthermore, for maximal performance, the hosting container should leverage a GPU for fast computations.

Success

Our work is the first to use Stanford backgrounds for artistic style transfer and one of the first to combine automatic segmentation with neural style transfer.

Future Work

While the first of its kind semantic style transfer has shown remarkable aesthetic results, improving the segmentation algorithm on a multifaceted collection of images beyond simple backgrounds can unlock better segmentation models. Furthermore, teaching the model to handle more sophisticated backgrounds can produce sharper segments, where a mixture of styles are not distorted into the background as a result of noise. The app can also allow users to provide their own unique styles for greater customizability and offer a more tailored experience.

7. CONCLUSION

All in all, we were able to successfully assemble an app that provides users with more freedom in styling their images by way of automatically segmenting out regions of interest that they can apply a style to. Results show that both the segmentation and style transfer models are able to work harmoniously to conjure perceptually pleasing results, especially on panoramic images.

8. TASK DISTRIBUTION AMONG MEMBERS

The task distribution among team members is tabulated as follows:

Name	Assigned Tasks
Sidharrth Nagappan	UNet semantic segmentation finetuning, Masked semantic style transfer, and App development
Nadia Ahmad Pirdaus	Foreground-background segmentation via Detectron2 with GrabCut edge refinement
Cornelius Pang	Fast style transfer using pretrained Magenta model, Masked semantic style transfer and App development

9. DECLARATION OF FYP

None of the members of this team are involved in an FYP that directly implements techniques used in this proposed project.

Name	Project Name
Sidharrth Nagappan	Document Scoring Using Natural Language Processing
Nadia Ahmad Pirdaus	Underwater Image Restoration
Cornelius Pang	Distributed Machine Learning

10. REFERENCES

1) Kyprianidis, J. E., Collomosse, J., Wang, T., & Isenberg, T. (2013). State of the "Art": A taxonomy of artistic stylization techniques for images and video. *IEEE Transactions on Visualization and Computer Graphics, vol. 19, no. 5, pp. 866-885.* doi: 10.1109/TVCG.2012.160.

2) Efros, A. A., & Leung, T. K. (1999). Texture synthesis by non-parametric sampling. *Proceedings of the Seventh IEEE International Conference on Computer Vision, 2, 1033-1038 vol.2.*

3) Gatys, L. A., Ecker A. S., & Bethge, M. (2016). Image style transfer using convolutional neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2414-2423.* doi: 10.1109/CVPR.2016.265.

4) Prisma Labs. (2016). *Styling solely the foreground or background of an image* [Screenshot]. Lensa. https://prisma-ai.com/

5) Ghiasi, G., Lee, H., Kudlur, M., Dumoulin, V., & Shlens, J. (2017). Exploring the structure of a real-time, arbitrary neural artistic stylization network. *British Machine Vision Conference (BMVC) 2017.*

6) Makow, N., & Hernandez, P. (2017). Exploring style transfer: Extensions to neural style transfer.