Comparative Paper of Style Transfer Models

Leonardo Seoane Navya Jain Jeya Varshini Bharath

1. Introduction

The last few years have resulted in remarkable progress for Deep Learning Models across almost every domain. There have been drastic changes in the performance of language models such as Chat GPT, image generation models like DALL-E which is able to generate an image from a text prompt, and audio based models such as Google's MusicLM which can generate songs from text. These new innovations will almost certainly have an impact on how we create art in the future. An almost infinite number of new songs, poems, screenplays, and songs can be created with assistance from an AI, without the user having an in-depth knowledge of painting, poetry, or song making. This paper focuses on style transfer models which are primarily concerned with modifying visual art. These models can mimic the "style" of an artist by taking an image, and imparting that particular style to create a new image. Effective implementations of these types of models could allow us to "remix" movies and art in new and unexpected ways, create new art based on deceased artists unique art style, as well as serve as aids to the long and arduous animation process. In this paper, we explore different implementations of style transfer models, examining which models work best, and experimenting with ways to achieve better results. In particular, we will examine two models, and two different input types. The first model we focus on is a classic VGG-19 based GAN model. The second model is the dual style GAN model which is the current state of the art model for style transfer. Finally, we experiment with the input to the models to see if preprocessing the training data results in more accurate style transfers. In particular, we examine if taking the edges or contours of a training image before passing it to the model increases style fidelity.

2. Datasets

Two datasets were used for the training, testing and validation of our style transfer models, the Celeb A dataset which contains over 200k images of celebrities from the shoulders up, and the Anime Celeb dataset, which contains over 63k images mirroring the Celeb dataset, but with different angles and poses of the characters. We took the first 10 images from the Celeb A dataset for testing the contourGAN. For extracting the image contour data, the PASCAL dataset containing over 10k images of a variety of different images and scenes was used for training,testing, and validation. The training subset we used for Pascal contained 6000 images, with 200 each for testing and validation. The AnimeCeleb dataset contained 3499 images in total. The dataset was split with a ratio of 80:10:10 for train, validation, and test sets respectively. Using this split ratio, there are 2879 images in the training set, 360 images in the validation set, and 360 images in the test set.The pre-trained StyleGAN generators are trained on the FFHQ dataset, which contains 70,000 high-quality human faces at 1024x1024 resolution. The mapping network is fine-tuned using the VGGFace2 dataset, which contains over 3.3 million images of faces.

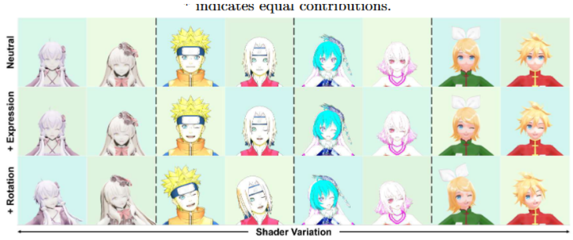


Figure 1. From AnimeCeleb Dataset

3. Methods and Details of the Approach

For extracting the image contours from the training set, four different approaches were used after some research. We found that neural networks could extract the image contours with high fidelity and produce an output with clear defined edges similar to those we hypothesize would improve training. Therefore, we first adapted the ContourGAN from H.Yang et al. The second approach is from J.Yang et al which only contains an encoder-decoder model where the information from pooling layers is stored and reused to unpool in the decoder. The third method is a hybrid between both papers, where we attempt to combine aspects of both. Finally, we used traditional linear Gaussian convolutions to extract the contours which serve as our baseline. For the first approach, the authors of H Yang et al. create a generator with a traditional U net-based encoder-decoder architecture, where every convolutional block in the encoder has a skip connection to its corresponding layer in the decoder.

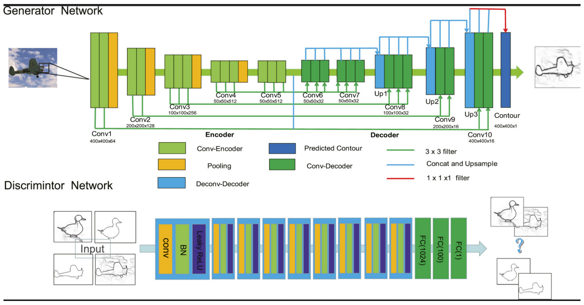


Figure 2. From H.Yang et al. model architecture for the Contour GAN. J. Yang et al.'s encoder-decoder network used a similar architecture for the encoder-decoder, but the skip connections are only at the pooling layers

The encoder part of the generator is a modified pre-trained VGG-16 model. For the decoder, the architecture is similar to the encoder, but we use bi-linear interpolation and transposed convolutional layers to upsample the model. The discriminator part of the GAN is a CNN containing nine layers of convolutional blocks trained to identify the ground truth training labels from the generator output. In our implementation of the ContourGAN, besides reducing the batch size, the weighted BCE loss, and the preprocessing steps we outline below, every detail of their implementation was preserved. For the hybrid implementation in the third approach, we attempted to use the architecture of J.Yang et al. for the generator, and kept the implementation of H. Yang et al's ContourGan for the discriminator, but modifying the discriminator by adding pooling layers at the end to reduce the computational complexity. The final approach, convolving the image with a Gaussian filter, is a quick and lightweight method that yielded good results. Testing on the celebA dataset showed it preserved key information such as facial features, but removed some details from the hair and background.

All models were trained on a subset of the PASCAL dataset for 15 epochs, using center cropping to 224 x224, image normalization, color jitter, and random horizontal flipping. Due to the unique approach that the authors suggest, there is not an abundance of literature and research to guide how to improve the model, but we modified the architecture and tuned hyper-parameters to see if performance would improve. Oftentimes for the two models with discriminators, the discriminator was too powerful and correctly identified every generated and ground truth image, resulting in 0 loss for many epochs. Layers of the convolutional blocks in the discriminator were removed, but it did not solve the issue. At times, the models did not always converge; and the training generator loss varied widely. We also experimented with adjusting the learning rate, adding weight decay to the optimizer, and varying the training epochs and batch size. Due to training constraints and memory requirements, we could not increase the batch size past a size of four, as opposed to H Yang et al. 's recommended batch size of eight. Because of this limitation, it is possible that each mini-batch was not large enough to effectively stabilize the loss during training. Furthermore, we experimented with modifying the regularization value placed on the adversarial loss when calculating the generator's loss by weighing the discriminator's decision less or more heavily. By lowering the alpha, we were able to extract better contours in all instances. Below, we will discuss the implementation of the style transfer models in more detail, starting with the VGG-19-based model.

To recap, our VGG-19 model implements a neural style transfer algorithm, which combines the style of one image with the content of another image to create a new image. This is achieved by optimizing a generated image that matches the content of the input image and the style of the reference image.The algorithm utilizes a pre-trained deep convolutional neural network (CNN), typically VGG-19, to extract the content and style features from the input and reference images. The content features are obtained by passing the input image through the CNN and extracting the output of a specific layer, while the style features are obtained by computing the Gram matrix of the output of several layers of the CNN. The optimization is performed using gradient descent, where the input image is iteratively updated to minimize a loss function that measures the difference between the content and style features of the input and reference images. The loss function is a weighted sum of the content and style losses, where the content loss measures the mean squared error between the input and content features, and the style loss measures the mean squared error between the Gram matrices of the input and style features. The algorithm also sets the style and content weights to balance the contribution of the two losses. This is a tunable hyperparameter that determines whether the model focuses more heavily on mimicking the style of the image or fitting the content image more closely.

The final approach discussed in the scope of the project uses the concept of Dual StyleGANs. In a nutshell, it takes two inputs to combine the image and its style to create an output with style transfer instead of taking only one image as an input.Introduced by Yang et al. (2022), Dual StyleGANs is a technique that involves training two separate Generative Adversarial Networks (GANs) on different styles of images and then combining the outputs of these two networks to generate a new image with a hybrid style. The original StyleGAN algorithm was developed by Nvidia and is widely used for generating high-quality, realistic images.The training steps involve first selecting two styles of images to be combined. Then, two separate GANs are trained using the StyleGAN architecture, one for each style. After training the two GANs, they are combined to generate new images that combine the styles of both datasets. This is typically done by passing a random noise vector through both GANs and then blending the output images together in some way.The code is an implementation of the DualStyleGAN algorithm proposed in the paper ”Dual Style Transfer for Portrait Images” by Yang et al. The algorithm enables the transfer of different styles from two different reference images onto a target image. Specifically, the algorithm uses two pretrained StyleGAN generators (one for each reference style), and trains a mapping network to map the intermediate latent codes of the target image to the intermediate latent codes of the reference images. The final output image is then generated by passing the mapped intermediate codes to the corresponding StyleGAN generators.The code uses two pre-trained StyleGAN generators, one for each reference style. The generators are trained on the FFHQ dataset, which contains 70,000 high-quality human faces at 1024x1024 resolution. The generator checkpoints can be downloaded from the StyleGAN2 repository.In addition to the pre-trained generators, the code also trains a mapping network to map the intermediate latent codes of the target image to the intermediate latent codes of the reference images. The mapping network is trained using a combination of the L1 loss and the perceptual loss, and is fine-tuned using the VGGFace2 dataset.Overall, the algorithm enables the transfer of different styles from two different reference images onto a target image. The code uses two pre-trained StyleGAN generators, one for each reference style, and trains a mapping network to map the intermediate latent codes of the target image to the intermediate latent codes of the reference images. The mapping network is trained using a combination of the L1 loss and the perceptual loss, and is fine-tuned using the VGGFace2 dataset. The final output image is generated by passing the mapped intermediate codes to the corresponding StyleGAN generators.

4. Results

For the image contour extraction approaches, our best models were able to achieve an average BCE loss of 0.3594135517254472 with just the encoder decoder model, resulting in reasonably enhanced contours. Our modified GAN, achieved an average BCE loss of 0.36375651067122816 and with this output. In both instances, we were able to enhance the existing ground truth contours, but there was not a meaningful difference between approaches. Finally, using the best approximation of the Contour GAN paper, we achieved an accuracy of 0.3836441689264029, but achieved the most visually appealing results. We are not overly concerned with the BCE loss, as the main metric which we believe is important is the visual component.

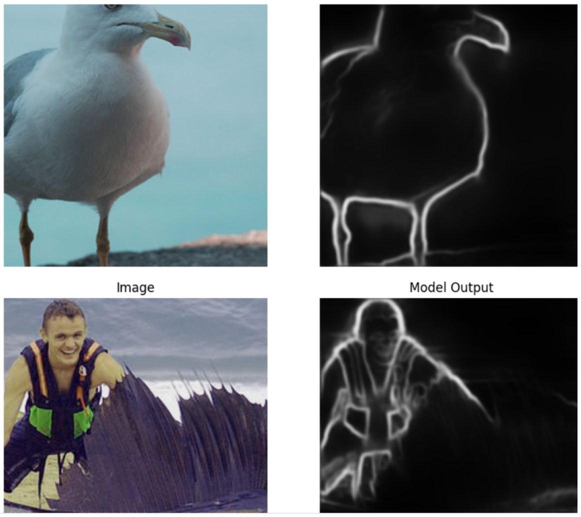
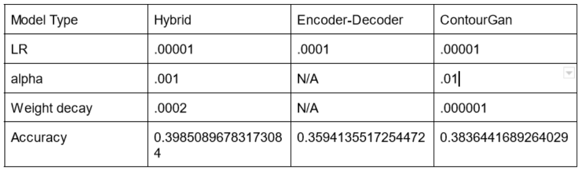


Figure 3. Best output using our implementation of ContourGan. The model is successfully able to extract most of the contours, but some details are missing, particularly edges within the edge boundaries



Model Type	Hybrid	Encoder-Decoder	ContourGan
LR	.00001	.0001	.00001
alpha	.001	N/A	.01
Weight decay	.0002	N/A	.000001
Accuracy	0.39850896783173084	0.3594135517254472	0.3836441689264029

Figure 4. Table showing our best results we obtained using our three different models and their corresponding hyperparameters.

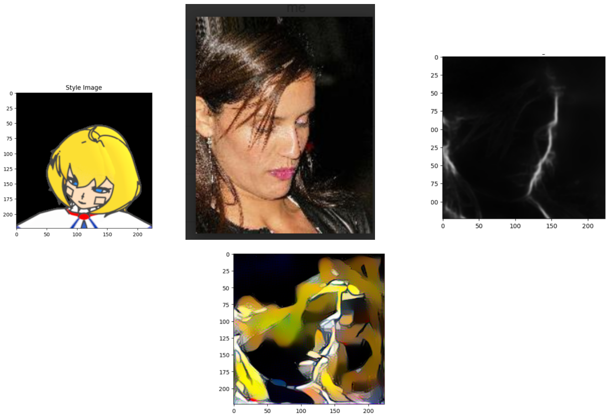


Figure 5. The style anime image, along with the context image. For the bottom images, we selected an image with more facial contours to see if performance improved,which it did.

Overall, the neural style transfer algorithm based on the VGG-19 backbone provides a powerful tool for creating artistic images by combining the content and style of different images. By using a pre-trained CNN to extract the features of the input and reference images, the algorithm is able to transfer the style of the reference image to the content of the input image, producing a new image that captures the essence of both. It is relatively easy to implement be-

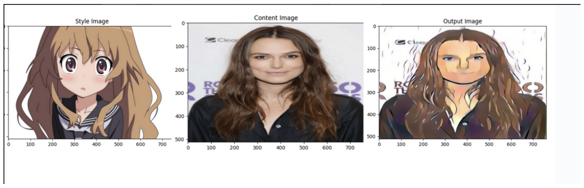


Figure 6. Table showing our best results on the fine tuned model we obtained and their corresponding training losses and runs.

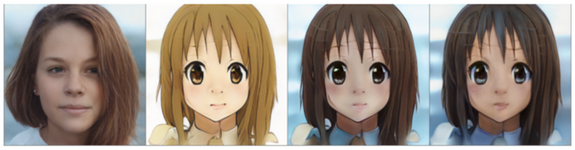


Figure 7. From left to right: 1. Input image 2. Style transfer 3. Structure transfer result: preserve the color of the content image by replacing the extrinsic color codes with intrinsic color codes 4. Structure transfer result: preserve the color of the content image by deactivating color-related layers

cause the backbone is a widely used and available model with little changes needed to achieve a reasonable result. However, this approach also has several limitations, and is far from the state of the art.

5. Discussion and Conclusions

We were successfully able to replicate several papers and extract high fidelity edge information from our dataset, and pass that information into our models.To test if these contours were able to improve the style transfer quality, we extracted the image contours from our celebA dataset and passed them to the VGG model.Unfortunately, there was not a marked improvement by passing the style transfer model contour images. Although we hypothesized that extracting the image contours would allow the model to focus on more relevant features, this did not end up being the case in practice. Many times, not enough facial information was preserved in the contour extraction process, resulting in dark spots where the person's face should be and obscuring the face. Interestingly, preprocessing the image by extracting the contours created more accurate hair color, something which the baseline model incorrectly predicts. Experimenting with this discovery is a promising area for future work. Additionally, in instances where there was more facial features present after the contour extraction, the model was able to generalize more effectively, which is shown below, even rounding out the face to mirror that of the style image. We hypothesize that if we trained the contourGAN on a dataset more sensitive to facial features, we could obtain results that outperform the current baseline model. Although using the image contour data did not enhance the stylizing effect, we hypothesize that some type of preprocessing step is necessary to achieve the best results.Further work needs to be done with training a model from scratch on the image contour data, which might allow the stylization process to run more smoothly and style and context to be better preserved. For the VGG-19 backbone GAN, one approach to improve the model would be to use transfer learning to fine-tune the model on a larger and more diverse dataset of images, which could improve its ability to generalize to new styles and image types. Another approach could be to experiment with different loss functions or optimization techniques, such as using adversarial losses or exploring different gradient descent methods. Additionally, we could explore ways to make the model more efficient, such as by using pruning or compression techniques to reduce the number of parameters in the network.We employed a single style image and a single content image for style transfer. However, it was found that trying different anime characters could result in mixed outcomes. An important limitation of using the VGG-19 model for style transfer is its large number of parameters, which leads to slow computation times. As a result, this model may not be suitable for high-resolution images or real-time applications. Additionally, the VGG-19 model was trained on a specific set of images, and its ability to generalize to other styles and image types may be limited.We attempted various techniques to improve the performance of the VGG-19 model. One of the approaches was to train the model on the entire style dataset, but due to hardware limitations, this was not feasible. Similarly, training on a smaller subset of the dataset was also attempted, but with limited success. Despite several other attempts to adjust the style and content weights, change the optimization algorithm, and modify the VGG-19 architecture, the model's performance remained suboptimal. Further experimentation is required to improve its efficacy.The project was limited by the available computational resources, which restricted effective model training and fine-tuning. If more time were available, transfer learning could be used to fine-tune the VGG-19 model on a larger and more diverse image dataset. Additionally, other optimization techniques and loss functions could be explored, including adversarial losses and different gradient descent methods.Furthermore, it may be worthwhile to investigate ways to increase the model's efficiency, such as parameter pruning or compression techniques, to reduce the number of network parameters. We also looked at two different methods for style transfer in depth, the VGG-19 based GAN style transfer model and the Dual Style GAN, which is the current state of the art to see the benefits and drawbacks of each model. While these models can serve as powerful tools to create new and interesting twists on existing forms of media, and benefit artists and casual users alike, it is important to keep in mind that there is potential for abuse. In many cases, an artist's sound/image/ art style is unique to them, and it is how they are able to distinguish themselves and make a living. Many times, years and years of practice and honing their craft is needed to develop an artistic identity. Abusing these models by not giving credit to the original author, or using this art style for profit without the consent of the author is something that legislators, artists, and machine learning researchers will have to discuss further to ensure that these tools are used to enhance art and expression, and not to detract from it. In the future, we are excited to continue work and research in this field, and explore all of the wonderful possibilities that come with this technology.

6. Team Contributions

Leo wrote the introduction, discussion, dataset, and conclusion sections of the report. He also was responsible for implementing the versions of the ContourGan,Hybrid GAN, and encoder decoder model papers outlined in the report and everything concerning extracting the image contours and using them in the VGG-19 style transfer model. He also contributed heavily to the project's organization and planning, compiling the final deliverables, and formatting and editing the final presentation and report.

Varshini was responsible for implementing the VGG-19 style transfer model outlined in the report and everything concerning the actual style transfer process. She also selected the content and style images used in the experiments, and fine-tuned the hyperparameters of the style transfer algorithm. In addition, she documented the implementation process and provided visualizations of the results.

Navya was responsible for working on the Dual StyleGANs implementation of the code.She also helped in refining the report and worked on writing her parts in the result and methodology.She also got access of the Anime dataset and created subsets to be used for the entire scope of the project.

References

Hongju Yang, Yao Li, Xuefeng Yan, Fuyuan Cao, ContourGAN: Image contour detection with generative adversarial network, Knowledge-Based Systems, Volume 164, 2019, Pages 21-28, ISSN 0950-7051, https://doi.org/10.1016/j.knosys.2018.09.033. (https://www.sciencedirect.com/science/article/pii/S0950705118304842)

Xie, Saining and Zhuowen Tu. ”Holistically-nested edge detection.” Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1395-1403. Yang, Jimei, et al. ”Object Contour Detection with a Fully Convolutional Encoder-Decoder Network.” arXiv preprint arXiv:1603.04530, 2016, https://doi.org/10.48550/arXiv.1603.04530.

https://github.com/captanlevi/Contour-Detection-Pytorch https://github.com/ylf-li/ContourGAN https://pytorch.org/tutorials/beginner/dcgan$_{f}aces_{t}utorial.htmlhttps$: $//www.programmersought.com/article/50936088861/https$: $//www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.htmlbsds500$

https://www.tensorflow.org/datasets/catalog/celeb$_{a}https$: $//www.ecva.net/papers/eccv_{2}022/papers_{E}CCV/papers/136680405.pdfhttp$: $//host.robots.ox.ac.uk/pascal/VOC/https$: $//stackoverflow.com/questions/66678052/how-to-calculate-the-mean-and-the-std-of-cifar10-data$

AnimeCeleb dataset - https://github.com/kangyeolk/AnimeCeleb CelebA dataset - http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html https://github.com/symoon94/Neural-Style-Transfer-pytorch/tree/master

Link to Dual GAN paper - https://arxiv.org/pdf/2203.13248.pdf Github for Dual GAN - https://github.com/williamyang1991/DualStyleGAN/tree/main

Leo also used the PS2 answer sheet code for a basic training template for the training and validation functions for the contour GAN models, and PS3 answer sheet code for extracting the image contours using gaussian kernels.