Human-to-Anime Style Transfer

Ciardo Roberto
Politecnico di Torino
Corso Duca degli Abruzzi, 24, Torino
s287279@studenti.polito.it

Rovegno Carolina
Politecnico di Torino
Corso Duca degli Abruzzi, 24, Torino
s306124@studenti.polito.it

Abstract

This work presents a machine learning technique for transforming an input image of a human face into an output image that closely resembles the characters from the Netflix TV series "Arcane". The proposed approach employs a conditional adversarial network (cGAN) that uses for the generator a ResNet50-UNet architecture, which is pretrained on the ImageNet. Additionally, a VGG16 model pretrained on the same dataset is used to compute a perceptual loss. The discriminator is a "PatchGAN" classifier that determines the realism of nearby image patches. While various generator architectures were considered in the experiments, the proposed ResNet50-UNet model demonstrated the best results based on both qualitative and quantitative metrics. In terms of quality, this network demonstrated the ability to generate high-quality Arcane-style images that are comparable to those obtained from the original ArcaneGAN. The visual quality is reinforced by the utilization of quantitative metrics, like FSIM and PSNR which effectively demonstrate the high quality of the image and the fidelity between the input image and the generated output.

1. Introduction

The use of machine learning models for image processing has become increasingly popular in recent years. Style transfer is a process that involves converting an input image into an output image with the same content as the original but with the style taken from another image. This technique has many potential real-world applications, including in the animation and graphics industry, where it can be used to create unique and personalized images that enhance the visual impact of various media.

To investigate the different types of networks and parameter configurations used to solve the general task of style transfer, this study reviewed related works[11], including recent networks[13] that have greatly improved the quality of generated images. CycleGAN[19], for example, proposes a cycle-consistency loss that removes the need to create a paired dataset, while Pix2Pix [8], which is trained on a paired dataset, uses a cGAN framework with an L1 loss to learn a mapping function from input to output images. This study also found a significant work based on ArcaneGAN [17][2] that achieved great results in the same task. However, the GitHub repository was not public, but some guidelines[17][5] on how the network was built were available. Based on these guidelines, an attempt was made to emulate ArcaneGAN in order to achieve results as similar as possible to the original network.

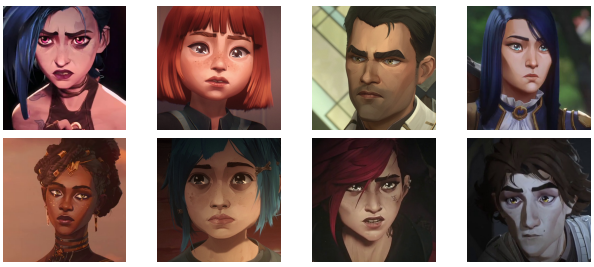


Figure 1: Examples of frames extracted from Arcane videos.

Using Pix2Pix[18] as the base framework, the first obstacle was the creation of the paired dataset. To overcome this, the character faces were extracted from the available Arcane episodes on Netflix and saved into a Frame Dataset. Figure 1 illustrates some examples of frames. Then, a StyleGAN2-ADA [13], pre-trained on real generated human faces[14][12][10], was trained on the frame dataset to generate new Arcane characters. By blending[4] the pretrained model with the newly trained network, a paired Arcane Dataset was generated. This dataset consists of images which have the content of the original human face image and the style of the Arcane characters, making it suitable for training the proposed model.

Following the architecture of the model reported in the GitHub repository[17] that used the *fastai v1 U-Net*, it was decided to start with the base Pix2Pix model[18] and modify it accordingly to handle our dataset, while also modify-

ing the architecture to match the dimensions of our images.

Instead of being directly trained, the generator models are trained by a second model called a discriminator, which learns to differentiate between real and fake or generated images. Therefore, there is no objective function or objective measure for the generator model, and the network applies manual inspection of the generated images, which is a good starting point to begin but is not an objective judgment. Quantitative measures have been further explored for our case study and can be combined with qualitative evaluation to provide a robust evaluation of the GAN models. Specifically, four different types of metrics were implemented: SSIM, FSIM, PSNR, RMSE.

After receiving unfavorable results from the metric analysis, modifications were made to the generator's architecture. The initial modification involved introducing nine residual blocks (U-Res9) [7]. Next, a configuration was tested that utilized a pre-trained ResNet50 (U-Res50) [6]. Finally, a pre-trained VGG16 was added to this architecture, enabling the calculation of perceptual loss [9] in order to improve the generator loss function and thus the image both geometrically and stylistically.

2. Dataset

Frame Dataset. The first step in creating this dataset was to collect the 9 episodes of the Netflix TV series "Arcane". 6 episodes were collected in 1080p and 5 episodes in 720p, reusing the same episodes in different qualities in order to increase the amount of data. The next objective was to extract the faces of the characters from individual frames of each video, using a Haar-Cascade human face classifier [3]. The Haar-Cascade Face Detection is a type of sliding window algorithm that detects objects based on their features, such as nose bridge, mouth line, and eyes. An OpenCV code was implemented which, given a single episode as input, checks for the presence of faces in each frame, selecting one and only one. By acting on the scaleFactor (1.1) and minNeighbors (4) parameters [1], we set the best configuration in order to achieve the right trade-off between correctly detecting the most faces and the fewest false positives. For each frame, the face was delimited by its bounding box, which initially was incorrect for the type of face to be cropped. After appropriate adjustments to the dimensions, which involve a percentage increase inversely proportional to the size of the bounding box, the frame was cropped. All crops were filtered to have a minimum size in pixels greater than 256x256, in order to maintain sufficient information for training the StyleGAN2-ADA network, which will be explained later. The crops were resized to a dimension of 512x512 pixels. Some detection defects were found in the image set, so a manual cleaning step was performed. The resulting frame dataset collects a total of 80,055 face images representing the limited characters of Arcane, viewed from different angles.

Train StyleGAN2-ADA. Given the limited number of characters obtained from the Arcane video series and the aim of generating a paired dataset, it is not feasible to use the frame dataset obtained in the previous step to address the task at hand. After careful research, it was discovered that Nvidia has created a family of neural networks, called StyleGANs [13], capable of generating high-resolution images of objects and human faces. The neural networks used by StyleGANs are particularly advanced and include both supervised and unsupervised learning techniques, as well as image processing methods such as style interpolation, which allows for the manipulation of the generated images' appearance. Among the various StyleGANs available, the decision was made to choose StyleGAN2-ADA (Adaptive Discriminator Augmentation) due to its several advantages over other StyleGANs. For instance, its greater robustness allows it to generate high-quality images even when there are limited data available. Another characteristic that drove the adoption of this neural network is its higher computational efficiency, which made it possible to use this family of networks given our limited resources.

Following the discovery of a code integrating the usage of StyleGAN2-ADA[13][16], it provided the starting point for training the Arcane generative face network. Considering the dimension factor of the Frame Dataset, it was decided to begin with a pre-trained network on human face images. Among the three possible options (256, 512, and 1024), the size of 512 was chosen, which matches the Frame Dataset. An attempt was made on the stylegan2-ffhq-512x512 network [14][12], but the results were unsatisfactory due to evident issues with the generation of eyes and teeth. Finally, the ffhq-512-avg-tpurun1 [10] was chosen, which produced good results in the early stages of training. 128 snapshots of the networks were generated, which were then used to test the blending of human faces with the faces of Arcane characters. The explanation follows.

Blending StyleGAN. Blending in Nvidia StyleGAN refers to a technique that gradually blends the features of two or more images to create an image that combines elements of each. In other words, blending enables the generation of an image that has some characteristics of one image and some of another, in a natural and realistic way.

In this work, the blending technique was used to transfer the style from the network created by us to the networks used as starting points. To achieve this, two different types of blending were tested, as shown in Figure 2. Both of them are based on the same concept, that is, the blending of layers from the two networks to take geometry from the first one and style from the second one. Specifically, geometry information is obtained from the lower layers of the network and style information from the higher ones.

The code that integrates StyleGAN2-ADA[13][16] also

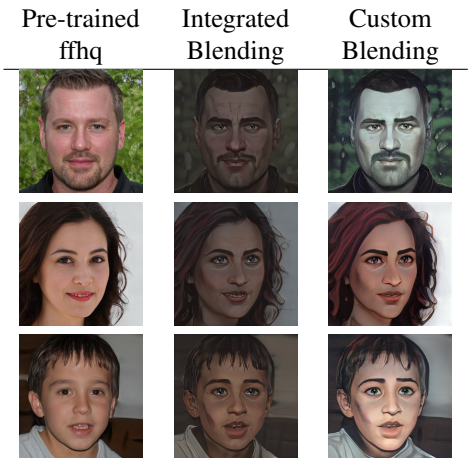


Figure 2: Examples of human face images processed using the two examined blending techniques.

allows for blending, which we will call Integrated Blending to differentiate it from the second type, which we will call Custom Blending. The latter was considered following the guidelines of the original ArcaneGAN. The two differ in the flexibility of combining the two networks. Integrated Blending involves defining a parameter that is used to determine the division point before the union is performed. We chose a value of 64 because it provides the best trade-off between correct geometry (higher values) and better style (lower values). Custom Blending, on the other hand, involves defining weights for each layer to more accurately control the combination of the two models. In this case, various configurations were tested and evaluated according to qualitative metrics until an optimal solution was reached.

For each snapshot of both generated networks, described in the Train StyleGAN2-ADA section 2, example images were created for comparison purposes. At this point, it was noted that the network pre-trained on stylegan2-ffhq-512x512 had problems generating eyes and teeth, resulting in opaque and lifeless or distorted features. In contrast, the ffhq-512-avg-tpurun1 network demonstrated the ability to faithfully reconstruct the original face while applying the correct style.

Further analysis revealed better performance among snapshots 32-48, and thus the two blending techniques were compared on these snapshots. The custom blending with parameters ['4':-0.2, '8':-0.2, '16':-0.2, '32':0.2, '64':0.5, '128':0.9, '256':0.7, '512':0.7,] was deemed to produce the most accurate results. Here, the ID field corresponds to the layer, and the VALUE field corresponds to the percentage of weights taken into consideration by the two networks.

Paired Dataset. Having chosen the blending network, a practically unlimited number of images could be generated, limited only by the generative capacity of the network itself. For this task, paired datasets consisting of 100 (for network testing), 10,000, and 50,000 different seeds were selected. As for the main dataset used in the various experiments, it consists of 10,000 images for the training set and 1,000 images for the test set.

3. Method

To perform the style transfer task, we utilized a pix2pix architecture based on an existing implementation [18] which generates images of building facades by utilizing the CMP Facade Database.

3.1. Generator

The initial implementation [18] is designed for 256x256x3 images and has been appropriately modified to accommodate the dimensions of the images in the dataset, which are 512x512x3. The pix2pix architecture employs a U-Net [8] structure comprising a sequence of downsample layers that progressively reduce the input image, sized 256x256x3, to a dimension of 1x1x512 through 2D convolutions. Correspondingly, an equal number of upsample layers are employed to restore the image to its original dimensions using transpose convolutions. These upsample and downsample layers are connected through skip connections, enabling the model to access information at various spatial scales and better address image translation challenges.

Encoder. The encoder component within the proposed architecture assumes a central role in extracting informative features from the input image. Its objective is to capture information across diverse spatial resolutions, thereby constructing a compact and meaningful representation of image characteristics. In our encoder implementation, we use a pre-trained ResNet50 model initialized with weights pretrained on the ImageNet dataset. This allows the model to benefit from the knowledge gained during training on a large set of diverse images. The encoder is composed of four convolutional blocks, each responsible for progressively extracting features at resolutions of 512x512, 256x256, 128x128, and 64x64, respectively, utilizing residual blocks. These blocks constitute an architectural paradigm that enhances training efficacy and the expressiveness of deep neural networks. By incorporating residual connections, which allow for the bypassing of one or more convolutional layers, these blocks facilitate the learning process. Subsequently, a crucial bridge is established between the encoder and decoder, ensuring seamless connectivity and facilitating the transfer of pertinent information for generating the final image.

Decoder. The decoder module plays a critical role in the image reconstruction process, responsible for translating encoded features into high-resolution output images. It employs various techniques, including upsampling, skip con-

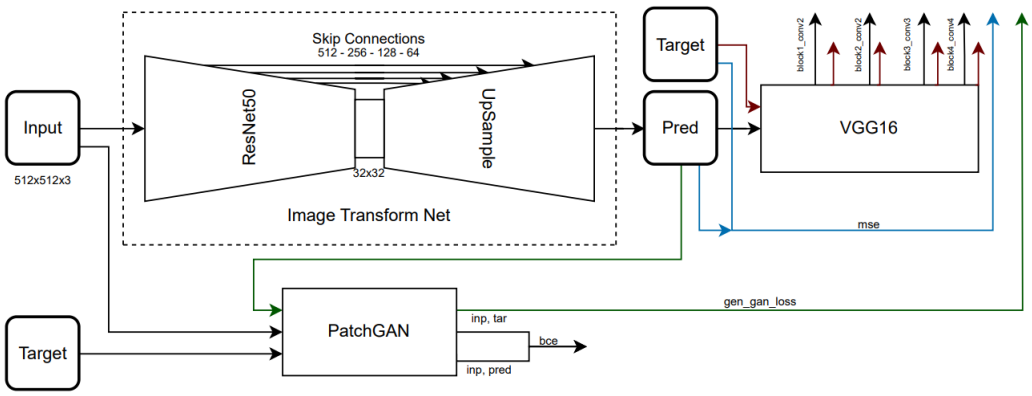


Figure 3: U-Res50-VGG16 network architecture. It consists of an image transform network that combines a pre-trained ResNet50 model and upsample layers from a U-Net. The generator's prediction and the target image are then processed by a pre-trained VGG16 network to compute perceptual loss. Additionally, the Mean Squared Error (MSE) is utilized as a metric to measure the dissimilarity between the target and predicted images. The PatchGAN discriminator evaluates the predicted image, input image, and target image using binary cross-entropy losses. These losses are combined to form the discriminator and the adversarial loss.

nections, and convolutional blocks, to restore fine details and spatial information. Upsampling is achieved through Conv2DTranspose layers, expanding feature maps to match the desired output resolution and recovering spatial structures lost during encoding. Within the decoder, convolutional blocks refine features using convolutional layers, batch normalization, and activation functions. These blocks shape learned representations and enhance the generator's expressive capacity. Activation functions, such as ReLU, introduce non-linearity, enabling the model to capture complex mappings between encoded features and the final output. To incorporate both local and global information, skip connections are utilized, establishing direct pathways between corresponding encoder and decoder layers. This facilitates information flow at different scales. By integrating low-level and high-level features, the generator leverages the contextual information captured by the encoder, resulting in improved image quality. Moreover, skip connections address the issue of vanishing gradients, where gradients diminish significantly during backpropagation across multiple layers. The direct connections established by skip connections enable smoother gradient flow between layers, thereby facilitating model convergence during training.

3.2. Discriminator

The discriminator used in this work is a PatchGAN model [8], designed to assess the quality of generated images compared to real images. It has been updated to support 512x512 images, while keeping the patch size unchanged. The input and target images are concatenated and undergo downsampling operations, creating a hierarchy of feature maps. Subsequently, convolutional layers and normalization are applied to generate the discriminator's prediction. Despite the increase in image size, the discriminator evaluates local features using the same patch size, ensuring consistent evaluation. This approach enables a detailed assessment of the generated images within the context of their local structures.

3.3. Training

Data. One of the initial challenges encountered with respect to the reference code was the approach employed for dataset importing. The ImageDataGenerator was employed to instantiate two instances, one for Arcane faces and the other for human faces, utilizing the flow_from_directory() function. The parameters batch_size=1 were specified to process a single image per iteration, shuffle=false, as reordering is unnecessary during this stage, and seed=1 to control the transformations.

These two generators were combined and passed as input to the Tensorflow Dataset function, facilitating the transformation into tensors of dimensions 1x512x512x3. Subsequently, a batch size operation was applied to the generated dataset, resulting in tensors of dimensions batch_sizex1x512x512x3. Following this, a function was applied to eliminate the batch size dimension introduced by the ImageDataGenerator, yielding tensors of dimensions batch_sizex512x512x3. Lastly, the images underwent a final transformation by applying data augmentation and normalization techniques.

Data Augmentation. In this study, data augmentation was randomly applied to the examples of the paired dataset

before being presented to the neural network for training. The data augmentation techniques used included rotation, cropping, and variations in contrast, saturation, hue, and brightness. The random application of these transformations allowed for the creation of a diverse set of similar yet distinct images, resulting in a more diversified dataset. Different color variations were applied to expand the chromatic range of the displayed images and enhance the visibility of specific details. This was necessary due to the peculiarities of the Arcane style, characterized by relatively low contrast and brightness, which could pose challenges during the network's learning phase. It is important to emphasize that the application of "on the fly" data augmentation techniques during model training did not involve any permanent modifications to the paired dataset. Each time an example is presented to the neural network for training, a random combination of data augmentation transformations is applied. This approach facilitate the generation of a continuous stream of diverse examples without the need for physical dataset expansion, thus effectively minimizing the computational impact.

Normalization. The chosen approach in this study involved normalizing the images to have values ranging from 0 to 1 in the described network. Before conducting the inference process for calculating the perceptual loss, both the generated images from the generator and the ground truth images were subjected to normalization using the input preprocess of the Keras application.

Hyperparameters. The focus of this study was to improve the performance of the networks. Different network configurations were explored, starting from the classical U-Net [15][8] architecture and progressing to U-Net with 9 residual blocks (U-Res9) [7] and a pre-trained ResNet50 model [6] initialized with weights from ImageNet. Subsequent modifications were made to the loss functions, including the incorporation of the Perceptual loss. All configurations were trained for 20 epochs, which represents the minimum required to evaluate the results given the limited computational resources available, which prevented further training iterations. The batch size was initially set to 1, which yielded the best results. Although attempts were made to increase the batch size, it was limited by GPU memory constraints, and therefore a batch size of 1 was maintained. Empirically, it has been shown that GANs tend to train better with a smaller batch size. The training process utilized the Adam optimizer, a variant of Stochastic Gradient Descent (SGD), with a learning rate of 2e-4. The choice of Adam optimizer was motivated by its demonstrated effectiveness in various GAN architectures. The alpha parameter of the Perceptual loss was adjusted to achieve improved results in the network training process. Mean Absolute Error (MAE) and Mean Squared Error (MSE) were used as evaluation metrics, specifically applied to the Perceptual loss. These metrics allowed for comparison and analysis, with MSE showing superior performance in this case. Different normalization techniques were applied to accommodate the requirements of different network architectures. Specifically, the U-Net and U-Res9 models utilized a normalization range of [-1, 1], while U-Res50 utilized a range of [0, 1]. These normalization ranges were found to be effective for the respective networks.

3.4. Loss

Perceptual Loss. The utilization of perceptual loss [9] in style transfer enables the generation of visually pleasing and coherent results, as it takes into account the relevant perceptual characteristics of the reference image without the need for a direct pixel-to-pixel comparison. This work presents an approach based on the utilization of a deep convolutional neural network, specifically the VGG16 network pretrained on ImageNet dataset. VGG16 is employed to extract feature maps at various levels of abstraction from both the input and reference images. These feature maps capture information about the structures, textures, and styles present in the images. To compute the perceptual loss, the feature maps of the input image and the generated image are compared using a loss function, such as mean squared error. The objective is to minimize the discrepancy between the feature maps of the input and generated images, aiming to generate an image that exhibits a similar style to the reference image. In the implemented approach, four feature maps are extracted from the VGG16 model by comparing the generated image with the ground truth image. However, the information regarding the difference in image content has not been taken into consideration.

Total Generator Loss. The total generator loss is composed of three main components: the adversarial loss, the perceptual loss, and the L1 loss. These losses are combined using specific weight parameters to achieve the desired balance in training the generative model. The adversarial loss is a measure of the discrepancy between the generated image and the ground truth image in terms of their realism. It encourages the generator to produce images that are indistinguishable from real images. The L1 loss, also referred to as the pixel-wise loss, measures the absolute difference between the generated image and the ground truth image at the pixel level. It encourages the generator to generate images with accurate pixel values and sharp details. To balance the importance of each loss component, two weight parameters are used: ALPHA and LAMBDA. LAMBDA is set to a value of 100, which reflects the significance of the L1 loss in our training process. ALPHA is a weight factor for the perceptual loss, which is carefully chosen between the range of 1 to 0.0001. The range mentioned allows for controlling the influence of the perceptual loss relative to the other loss terms. For the final configuration the value

0.001 was chosen. The total generator loss is calculated as the sum of the adversarial loss, weighted perceptual loss, and weighted L1 loss, as follows:

$$\begin{aligned} \text{total\_gen\_loss} = \text{gan\_loss} + (\text{ALPHA} \times \text{perc\_loss}) \\ +(\text{LAMBDA} \times \text{l1\_loss}) \end{aligned} \quad (1)$$

By optimizing this combined loss, we aim to train the generator to generate high-quality images that are both visually appealing and similar to the ground truth images in terms of content and pixel accuracy.

Training Loop.

Algorithm 1 Training Loop

```
Load test images
for each epoch in n epochs do
    for each batch in dataset_size // batch_size do
        Perform inference using the generator
        Perform inference using the discriminator on the target images
        Perform inference using the discriminator on the generator output
        Calculate generator loss based on steps 1 and 3
        Calculate discriminator loss based on steps 2 and 3
        Update the weights of the generator
        Update the weights of the discriminator
    end for
    Generate sample image
    Save logs and checkpoints
end for
```

During the training loop, a test step was also implemented to evaluate the network's performance on a validation dataset. However, validation data is not useful for monitoring loss or assessing issues like overfitting on the data. To assess the ongoing learning of the network, it is particularly important to ensure that the discriminator loss or adversarial loss does not approach zero, as it would indicate a "win" for one of the two parties and lead to network divergence. Throughout the training progress, the L1 loss and perceptual loss should decrease, so they were monitored to observe their decreasing trends.

3.5. Metrics

Four metrics are used to evaluate the performance of the proposed model: Peak Signal-to-Noise Ratio (PSNR), Root Mean Squared Error (RMSE), Structural Similarity Index (SSIM), and Feature Similarity (FSIM). Each metric offers unique advantages in assessing the quality and similarity of the predicted and ground truth images. These metrics collectively provide a comprehensive evaluation of the model's performance in terms of fidelity, perceptual similarity, and image quality.

PSNR. It is a widely adopted metric that provides a simple and intuitive measurement of image quality. It is computationally efficient and widely understood, making it easy to interpret and compare results across different models or algorithms. A higher PSNR value indicates a smaller difference between the clean and distorted signals, indicating a higher fidelity or better quality reconstruction.

RMSE. It offers a comprehensive measure of the average error between the predicted and ground truth images. It emphasizes larger errors, allowing for a more fine-grained analysis of model performance. Additionally, RMSE is differentiable, making it suitable for gradient-based optimization algorithms. Lower RMSE values indicate better similarity between the predicted and ground truth images.

SSIM. It evaluates structural similarities by considering luminance, contrast, and structural components. It mimics human perception and takes into account local image structures, making it particularly effective in capturing subtle differences and artifacts. SSIM also accounts for perceptual quality, which aligns with human visual perception. SSIM values range between -1 and 1, with values closer to 1 indicating higher similarity.

FSIM. It combines phase congruency (PC) and gradient magnitude (GM) features, providing a holistic assessment of perceptual similarity. By incorporating both high-level and low-level features, FSIM captures both fine details and global structure, making it suitable for evaluating the overall visual similarity of the predicted and ground truth images. FSIM values range between 0 and 1, with values closer to 1 indicating higher similarity.

4. Experiments and Results

4.1. Implemntation Details

U-Net. Based on the initial configuration of pix2pix [18], several modifications were made to the generator. The input layer was resized to 512x512x3. Additionally, a downsample layer was incorporated, utilizing a 2D convolution with 32 filters of size 4x4, employing "same" padding, a stride of 2. Similarly, an upsample layer was introduced to reverse the operation just described. The discriminator was also adjusted to align with the modified input dimensions, and a downsample layer was added accordingly.

U-Res9. Based on a referenced related work [7], the generator U-Net incorporated 9 residual blocks in its bridge phase. Each block follows a specific structure: two 2D convolutional layers with a 4x4 kernel size and "same" padding are applied. Batch normalization is utilized after each convolutional layer to ensure stable activations. A residual connection is established by summing the output of the second convolutional layer with the original input. The ReLU activation function is then applied to the final output of each

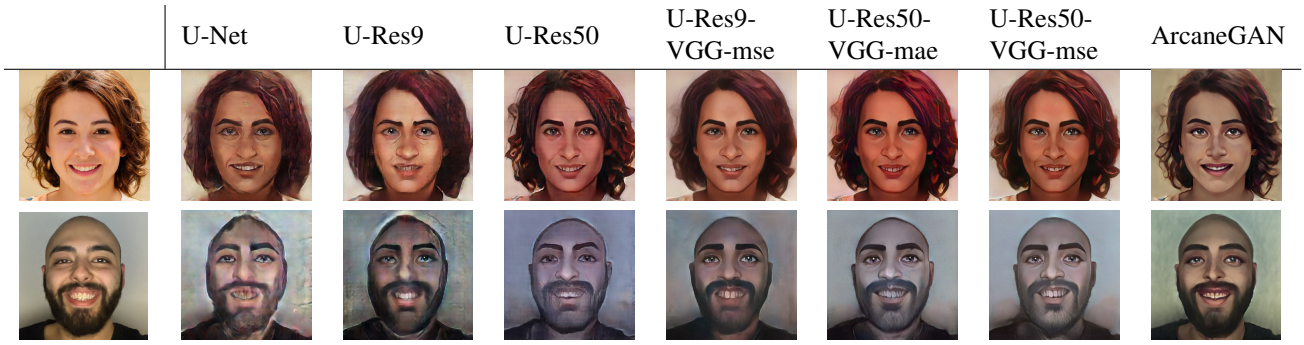


Figure 4: Images generated by the different networks evaluated after 20 training epochs. ArcaneGAN is the original network.

residual block. This mechanism facilitates efficient gradient propagation, enabling deep learning and improving the overall model performance.

U-Res50. In order to enhance the generator's capabilities, the guidelines provided in [17] [5] [7] were followed. To achieve this, a ResNet50 model pre-trained on the ImageNet dataset was incorporated, as mentioned earlier in the Generator section 3.1.

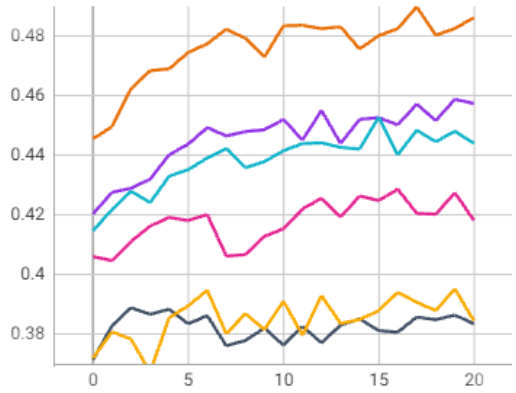
VGG16. Based on another case study [5] [9], the pre-trained VGG16 model on ImageNet was integrated into the U-Res9 and U-Res50 networks as an additional loss function. The purpose and functionality of incorporating the VGG16 model are described in detail in the Perceptual Loss section 3.4.

Comparison MSE/MAE. Following the integration of VGG16, the challenge of comparing the obtained feature maps emerged. To address this, two distinct metrics, MSE (Mean Squared Error) and MAE (Mean Absolute Error), were employed to evaluate the networks. While these metrics are theoretically expected to yield similar results, in practice the MSE metric showcased superior performance.

4.2. Qualitative Comparison

The qualitative comparison was conducted to assess the model's ability to generate visually appealing and stylistically consistent results. Throughout the experiments, a notable improvement in image quality was observed in Figure 4. In the initial networks tested (U-Net and U-Res9), issues related to image noise and fine geometric details of the human face were encountered. However, the color style was accurately captured from the early stages. To address the geometric challenges, a pre-trained encoder (U-Res50) was introduced, leading to superior qualitative outcomes. The encoder demonstrated enhanced recognition of facial edges with finer detail, although some artifacts remained visible around the mouth and eyes. Subsequently, the incorporation of perceptual loss allowed for the capture of more intricate details at different network stages. A noticeable enhancement in the generated images is evident, particularly when comparing the eyes and teeth, which proved to be the most challenging areas to address.

Figure 5: Example of results obtained from the FSIM metric, based on training for 20 epochs.Higher values correspond to superior quality.



Grey : U-Net
Pink : U-Res50
Purple: U-Res50-VGG-mae
Yellow: U-Res9
Blue : U-Res9-VGG-mse
Orange: U-Res50-VGG-mse

4.3. Quantitative Comparison

To quantitatively describe the obtained results, various metrics (PSNR, RMSE, SSIM, FSIM) discussed earlier were employed. A representative test image from the test set was selected and evaluated for each training epoch across all networks. The resulting metrics were then plotted on graphs, with an illustrative example of FSIM shown in Figure 5. Due to limited computational resources, a single test image was used instead of conducting inference on the entire dataset and calculating average values.

Table 1 presents the results obtained from the different networks. Both U-Net and U-Res9 exhibited poor perfor-

mance in terms of image quality and geometric accuracy. However, the introduction of U-Res50 demonstrated noticeable improvements. Nevertheless, the most significant advancements were achieved with the incorporation of VGG16 as the perceptual loss function. Notably, the difference in utilizing MAE and MSE in the U-Res50 network with VGG16 is evident. Although the discrepancy in results is not substantial, the utilization of MSE yielded superior perceptual outcomes. This phenomenon can be attributed to the fact that even slight variations in FSIM and SSIM metrics correspond to noticeable differences in human perception.

Architecture	PSNR	RMSE	SSIM	FSIM
U-Net	18.66	29.76	0.107	0.383
U-Res9	18.25	31.19	0.126	0.384
U-Res50	18.86	29.07	0.169	0.418
U-Res9-VGG-mse	**20.38**	**24.42**	0.208	0.443
U-Res50-VGG-mae	18.48	30.37	0.211	0.457
U-Res50-VGG-mse	19.98	25.56	**0.242**	**0.486**

Table 1: Values of PSNR, RMSE, SSIM and FSIM metrics calculated on a single test image during a 20-epoch training

5. Conclusion

This study presents a GAN-based approach to address the task of image-to-image translation. Specifically, our networks generate an output image that resembles a character from the Netflix Series "Arcane" using a human face reference. A significant challenge encountered was the creation of a paired dataset, which was accomplished using the blending technique on StyleGAN2-ADA. While the overall outcome was deemed satisfactory, there is room for improvement in several aspects. Firstly, the acquisition of video frames from Arcane could benefit from enhancements in terms of contrast, lighting, image quality, and other relevant factors. Moreover, refining the frame selection process could minimize false positives and ensure the inclusion of facial expressions with greater diversity. This would increase the variety of the frame dataset, incorporating images with smiles, beards, or different angles, which could then be used to enhance the training performance of StyleGAN2-ADA. Furthermore, exploring different parameter configurations for blending could potentially lead to a better paired dataset. Another crucial aspect of this work was the improvement of pix2pix performance for style transfer. Various architectures were implemented to achieve such improvements, yielding pleasant results. Comparing the generated images with those of ArcaneGAN shown in the Figure 4, minimal discrepancies can be observed in terms of image perception. However, due to limited computational resources, we were unable to perform extensive fine-tuning experiments on the networks or train them for a large number of epochs, which could have potentially yielded state-of-the-art results. It is proposed to work with smaller-sized images to mitigate memory occupancy issues and reduce training times, which currently average around 1 hour per epoch. Additionally, passing the predicted image to a network capable of increasing its resolution can be considered.

References

[1] cv::CascadeClassifier Class Reference. https://docs.opencv.org/3.4/d1/de5/classcv_1_1CascadeClassifier.html. detectMultiScale.

[2] Demo ArcaneGAN. https://huggingface.co/spaces/akhaliq/ArcaneGAN.

[3] Face Detection using Haar Cascades. https://docs.opencv.org/3.4/d2/d99/tutorial_js_face_detection.html.

[4] StyleGAN network blending. https://www.justinpinkney.com/stylegan-network-blending/.

[5] adriang. GANs based on abbreviated Lesson 7. https://github.com/aarcosg/fastai-course-v3-notes/blob/master/refactored_by_topics/CNN_L7_gan_feature-loss.md, 2019.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[7] Jason Hu, Weini Yu, and Zhouchangwan Yu. Image-to-Image Translation with Conditional-GAN. *Stanford Univerity*, 2018.

[8] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.

[9] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *CoRR*, abs/1603.08155, 2016.

[10] justinpinkney. Awesome Pretrained StyleGAN2. https://github.com/justinpinkney/awesome-pretrained-stylegan2#faces-FFHQ-config-f-512x512, 2022.

[11] Bing Li, Yuanlue Zhu, Yitong Wang, Chia-Wen Lin, Bernard Ghanem, and Linlin Shen. Anigan: Style-guided generative adversarial networks for unsupervised anime face generation. *CoRR*, abs/2102.12593, 2021.

[12] nvidia. nvidia models. https://api.ngc.nvidia.com/v2/models/nvidia/research/stylegan2/versions/1/files/, 2021.

[13] NVlabs. StyleGAN2 with adaptive discriminator augmentation (ADA) — Official TensorFlow implementation. https://github.com/NVlabs/stylegan2-ada, 2021.

[14] NVlabs. StyleGAN2 — Official TensorFlow Implementation. https://github.com/NVlabs/stylegan2, 2021.

[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[16] Derrick Schultz. StyleGAN2-ADA-PyTorch. https://colab.research.google.com/github/dvschultz/stylegan2-ada-pytorch/blob/main/SG2_ADA_PyTorch.ipynb, 2022.

[17] Sxela. ArcaneGAN. https://github.com/Sxela/ArcaneGAN, 2022.

[18] TensorFlow. pix2pix: Image-to-image translation with a conditional GAN. https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/generative/pix2pix.ipynb#scrollTo=_xnMOsbqHz61, 2019.

[19] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017.