Synthesizing High-Quality Face Images from Segmentation Maps or Sketches CSE575: Group 15

Yashas Puttaswamy
yputtas1@asu.edu

Vedant Parthesh Joshi
vedantjoshi@asu.edu

Raghavendra Dinesh
rdinesh2@asu.edu

Monisa A. Arivalagan
marivala@asu.edu

Abstract

We present pixel2style2pixel (pSp), an advanced image-to-image translation system, and show how well it produces face pictures from sketches and segmentation maps. In contrast to earlier techniques, pSp can accommodate partial or sparse data and is not dependent on pixel-wise correspondence. Pixel2style2pixel architecture is built on a unique encoder network that produces a stream of style vectors that are then fed into the StyleGAN generator that has been trained to produce the extended W+ latent space. We tried to demonstrate that with no further optimization, this encoder can immediately incorporate actual pictures into W+ space. The solution can perform a variety of jobs even when the input picture has no representation in the StyleGAN domain by straying from the conventional "invert first, edit later" practice utilized with earlier StyleGAN encoders. We demonstrate that pSp is capable of producing more varied and in-depth results when compared to state-of-the-art methodologies. We used CelebA-HQ dataset for sketch-to-image synthesis because there weren't any publicly accessible datasets. We demonstrate that pSp is capable of producing high-quality outputs across a range of input postures and emotions. The multi-modal approach renders it simple to create several outputs with the same pose and characteristics but different styles.

1. Introduction

In the field of computer vision and deep learning, significant tasks include facial segmentation and image-to-image translation. Face editing, facial trait transfer, and virtual try-on are a few uses for these jobs. The pixel2style2pixel (pSp) framework is an image-to-image translation technique that uses Generative Adversarial Network (GAN) inversion methods and latent space manipulation to produce high-quality images. The extended W+ latent space produced by feeding a collection of style vectors into an encoder and then sending it to a StyleGAN generator that has already been trained forms the basis of the pSp architecture. This method offers a simple replacement that is more suited for incomplete or sparse sketches, in contrast to conventional approaches that involve strict limitations demanding pixel-wise consistency between input sketches and output pictures. On the Face From Sketch and Face from Segmentation Map tasks, two facial image-to-image translation tasks, we assess this methodology.

We illustrate that, in the Face From Sketch problem, this method offers a replacement to conventional approaches that call for stringent requirements for pixel-wise correlation between an input sketch and output image. Since there is no publicly accessible dataset that accurately represents sketches, we create a customized dataset of face sketches and face features along with corresponding images from CelebA-HQ dataset. We test utilizing pSp for generating face pictures from segmentation maps in the Face from Segmentation Map problem. On the CelebAMask-HQ dataset, which has 19 semantic categories, we present an illustration of the other approaches. This method can produce excellent results from a wider variety of inputs with diverse views and expressions. Furthermore, for one input, such as a semantic map or sketch image, pSp can quickly produce a variety of achievable outputs with similar poses and attributes but different fine styles using the multi-modal technique.

This method may be used for a variety of different image-to-image translation problems in addition to the human face realm. Our findings show that pSp is a less complicated and more efficient substitute for conventional techniques that need rigid restrictions on pixel-wise correspondence. Additionally, this method can produce a variety of high-quality outputs from a wider range of inputs and contributes to multi-modal synthesis by resampling styles.

2. Project Description

2.1 Goal and Objective:

Gaining a thorough grasp of Generative Adversarial Networks (GANs) and their potential uses in picture production was the aim of our endeavor. We specifically concentrated on the Pixel2style2pixel (pSp) framework, a cutting-edge approach for picture synthesis, and sought to assess its performance in comparison to other approaches.

To do this, we first performed a detailed analysis of the pSp framework, carefully going over the study article on pixel2style2pixel. We evaluated the architecture and algorithm of the framework to determine its advantages and disadvantages. To ensure a comparative analysis, we then looked into other strategies and existing methodologies. We examined methods like pix2pixHD, and ALAE to compare the performance of the pSp framework to other cutting-edge models.

We put up an experimental environment using PyTorch and other relevant libraries to conduct our testing. The Pixel2style2pixel (pSp) framework and a StyleGAN encoder were painstakingly constructed. This process took more than four weeks and several stages. Additionally, we created our own sketches and maps to create the dataset from the CelebA-HQ dataset for our project. For our effort, we investigated several datasets in addition to the CelebAMask-HQ dataset that was utilized in the paper. After setting up our experimental environment and implementing the pSp framework, we trained the model using the CelebAMask-HQ dataset. We assessed the pSp framework's effectiveness using the datasets and examined its ability to handle a variety of inputs, including a range of facial expressions.

Overall, the goal of our project was to learn more about GANs and how they may be used for picture synthesis. We aimed to contribute to the field of computer vision and enhance the state-of-the-art in picture generation by investigating the pSp framework and evaluating its performance in comparison to other methods.

2.2 Contribution:

Yashas Puttaswamy: He thoroughly understood the architecture of pSp and StyleGAN and how pSp sends the encoded vector to StyleGAN. Additionally, he implemented the FusedLeakyReLU activation function and generated the necessary training and testing scripts required to train the model. Throughout the project, he worked on obtaining results for both models and created images to compare the ground truth, maps/sketches, and the output of the model.

Vedant Parthesh Joshi: He shared the findings with the team, explaining the concepts in a concise manner. He also searched for datasets that could be used for training this model and concluded with the celebAMask-HQ Dataset. He applied Edge Detection and Thresholding techniques to convert real images in the dataset to sketches for training the model. Additionally, he researched alternate ways to train the model and eventually took care of training the pix2Style2pix model for the sketch-to-face task.

Raghavendra Dinesh: He researched on the pix2pixHD paper and its significant discoveries. He developed segmentation masks using individual facial features and image processing techniques. In his quest to find a suitable dataset for model development and training, he searched various resources and finally concluded with the celebAMask-HQ dataset. He generated the masks dataset from the CelebA-HQ dataset and tested a few images to ensure the feasibility of training the model.

Monisa A. Arivalagan: She shared the findings with the team, explaining the concepts in a clear and concise manner. She examined the ALAE paper and identified its significant discoveries, adding it to the report. She took care of training the pixel2Style2pixel model for the map to face task and generated the loss graph from the logs obtained during training. She worked on the proposal report, milestone and final report writing and understood each other's strengths, deciding on tasks accordingly.

3. Approach

3.1 Literature Survey and Background:

In this section, the key points that were taken while going through the paper are discussed

The paper discusses three techniques related to GANs: GAN inversion, latent space manipulation, and image-to-image translation. GAN inversion involves finding the latent vector used by a pre-trained GAN to generate a given image, which can be done through direct optimization or training an encoder. The authors propose a new encoder that can accurately and efficiently embed a face image into the extended latent space W+. Latent space manipulation involves editing the latent code of an image to achieve semantic changes. Various methods have been proposed, such as finding linear directions for binary labeled attributes, utilizing pre-trained 3DMM, and finding latent space paths for specific transformations. The authors' approach encodes input images directly into output latent, allowing for handling inputs that are outside the StyleGAN domain. Image-to-image translation involves learning a function that maps an input image to a corresponding image of a different domain. The authors' method uses the same architecture to solve various tasks, making it more versatile compared to previous works that require dedicated architectures and do not generalize to other translation tasks.

Pixel2Style2Pixel Framework: The pSp framework utilizes a pre-trained StyleGAN generator and the W+ latent space. To use this representation, a strong encoder is needed to accurately match each input image to an encoding in the latent domain. In pSp, they extend the encoder backbone with a feature pyramid, generating three levels of feature maps from which styles are extracted using an intermediate network called map2style. The styles, aligned with the hierarchical representation, are then fed into the generator in correspondence to their scale to generate the output image, completing the translation from input pixels to output pixels through the intermediate style representation. The output of the model is defined as pSp(x):=G(E(x)+w), where E(ů) and G(ů) denote the encoder and StyleGAN generator, respectively. This formulation aims to learn the latent code with respect to the average style vector, resulting in better initialization.

Loss Function A latent style vector is produced by the encoder using a loss function based on four significant losses, which may then be used by StyleGAN to create an image. The encoder was trained using the following loss functions:

Pixel-wise L2 loss: This gauges the variance in individual pixels between the output and input images.

$$L_2(x) = ||x - pSp(x)||_2 \tag{1}$$

LPIPS:LPIPS loss compares the features that a pre-trained network collected from the input image and the produced image to determine how similar they are perceptually.

$$L_{LPIPS}(x) = ||F(x) - F(pSp(x))||_2 \tag{2}$$

Regularization loss: This promotes the encoder's output latent style vector to be more like the typical latent vector.

$$L_{reg}(x) = ||E(x) - \overline{w}||_2 \tag{3}$$

A weighted sum of these four losses is referred to as the total loss function, and the weights might vary depending on the encoding task at hand.

Model's Architecture: In summary, the model architecture is the base pSp architecture with a new activation layer which is an encoder based on a Feature Pyramid Network where style vectors are abstracted from different pyramid scales and inserted directly into a fixed pre-trained styleGAN generator in correspondence to their spatial scales. The following figure represents the network architecture that we followed.

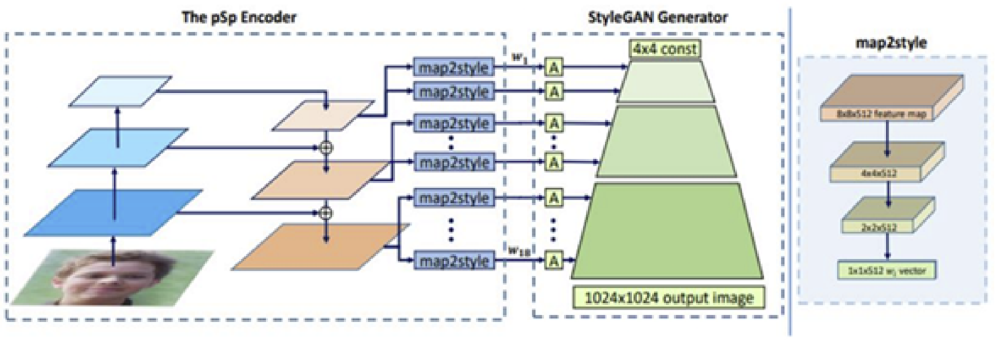


Figure 1: Architecture of the Model

The architecture of the PSP framework can be understood through a fundamental mathematical expression:

$$PSP(X) = G(E(x) + w) \tag{4}$$

This expression represents the transformation of an input image X through an Encoder block E, which extracts and encodes its features into a latent space, followed by the addition of a random noise vector w. Finally, the transformed latent space is fed into the StyleGAN Generator G to generate the output image.

3.2 Comparative study:

pix2pixHD: The pix2pixHD architecture consists of a generator network and a discriminator network. The generator network takes in a low-resolution input image and produces a high-resolution output image. It consists of an encoder-decoder architecture with skip connections between the encoder and decoder layers to retain spatial information. The encoder network is composed of a series of convolutional layers, while the decoder network consists of a series of transposed convolutional layers. The discriminator network is a PatchGAN architecture that outputs a binary classification score for each patch of the input image. It is used to distinguish between the generated and real images. The generator network is trained using adversarial loss, which encourages the generated images to be indistinguishable from real images, and a feature matching loss, which encourages the generator to produce images that match the intermediate features of the discriminator network. Additionally, the pix2pixHD architecture includes a multi-scale discriminator that evaluates the generated images at different scales to better capture the overall image structure.

In summary, the pix2pixHD architecture is designed to generate high-resolution images from low-resolution inputs by using an encoder-decoder architecture with skip connections, and is trained using adversarial and feature matching losses with a multi-scale discriminator. Since Pix2pixHD's architecture was not created with this purpose in mind, it might not be the ideal option for converting segmentation mask to facial images. To transfer the input segmentation mask to a matching face image, the Pixel2style2pixel approach uses an encoder network and a StyleGAN generator. It was created particularly for this application. In order to direct the StyleGAN generator to create a related facial picture, the encoder network is made to extract the appropriate style vectors from the input segmentation mask. It is a difficult challenge for Pix2pixHD, but this approach enables

Pixel2style2pixel to build very realistic facial pictures using segmentation masks with fine-grained features.

ALAE: Adversarial Latent Autoencoder, or ALAE for short, is an autoencoder designed to overcome some of the drawbacks of generative adversarial networks. The two primary components of the design are learning the output data distribution using an adversarial technique and learning the latent distribution from the data to overcome entanglement. By doing this, ALAE integrates recent developments in the field with the generative capabilities of GANs. To prevent less-than-ideal outcomes from happening in the picture space, ALAE leverages AE reciprocity to impose the latent space rather than the data space for reconstruction losses. As a result, ALAE is better able to generate visual details while maintaining the generative capabilities of GANs. When compared with pix2Style2pix these are the key differences we found: pSp is a conditional GAN-based model that generates an output image given an input image and a target style, while ALAE is an unsupervised model that learns to map images from one distribution to another. pSp operates in a disentangled latent space, allowing for easier manipulation of individual image attributes, while ALAE operates in a continuous latent space, providing greater flexibility in image-to-image translations.

4. Methodologies

4.1 Implementation:

Sketch dataset Generation: We created a dataset of sketches from an image dataset using OpenCV's Canny edge detection algorithm. Several methods were used to turn an image into a sketch. The cvtColor function is used to convert the picture first from RGB to grayscale, creating a single-channel image where each pixel's value corresponds to its brightness. The grayscale image is then subjected to Gaussian blur using the Gaussian blur function, which averages the values of pixels near each pixel, to minimize image noise. The edges of the picture are then located using Canny edge detection. This entails locating the intensity gradients in the picture and highlighting the areas with the sharpest gradients. Finally, to create black lines on a white backdrop, the image is inverted using the bitwise not function.

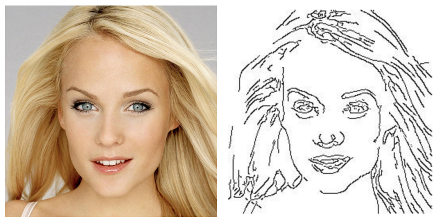

(a) Original Image (b) Sketch

Figure 2: Generating Sketch dataset

Segmentation maps dataset Generation: The CelebAMask-HQ dataset contains high-resolution images of faces with up to 19 segmentation masks for various facial features for each real image. Each segment image is grayscale. We have to overlap the images to generate segmentation masks with each image of pixel values ranging from 0 to 18 and each value corresponds to a specific facial feature such as eyes, nose, mouth, skin, etc. Once we loaded all the segment images, we overlapped them to create the segmentation mask for that real image. This can be done by assigning the pixel values of each segmented image to the corresponding pixels in the final segmentation mask. For example, if the pixel value for skin in the skin segment image is 1, you can assign all pixels in the final mask with the value 1 to represent the skin. Refer to Figure 2 for examples of individual facial feature images and the resulting overlapped mask image.

Transfer Learning: Transfer learning was necessary primarily because the images returned from the pretrained model for both map to face and sketch to face did not correspond to the real-world photos. Further examination revealed that the generated facial images missed several of the details from the sketches or maps, such as improper visibility of the eye boundaries. Poor performance on these tasks was a result of this. Another explanation is that the datasets distribution was altered because the image processing techniques utilized to construct it were different from those described

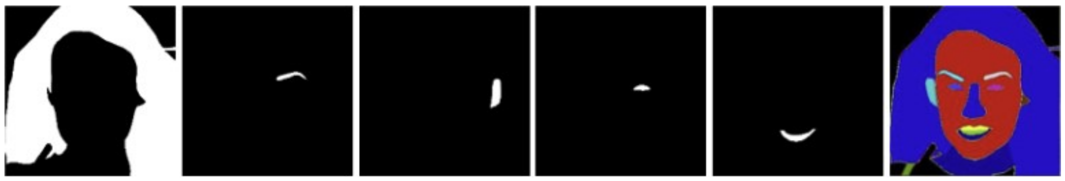


Figure 3: Segmentation Masks

in the paper. Because of this, the pretrained model was unable to learn the characteristics unique to our dataset and was therefore unable to successfully use these features. Additionally, the model was unable to capture all the necessary elements for producing realistic photos, as evidenced by the similarity score between the output images and the ground truth images not matching the values in the study.

Transfer learning was essential for overcoming these obstacles. On our particular dataset, this involved fine-tuning a pretrained model that has already learnt pertinent characteristics from a big dataset, such as StyleGAN encoder. The model was able to learn the precise elements needed for creating realistic images from maps and sketches by applying transfer learning, which improved performance on these tasks.

Table 1: Common Hyperparameters for both models

batch size	6
device	Tesla T4
encoder type	GradualStyleEncoder
nput nc	19
l2 lambda	1.0
label nc	19
learning rate	0.0001
lpips lambda	0.8
optim name	ranger
test batch size	6
val interval	5000

Training Details: The training was done on Google Colab pro with 15GB GPU. the GPUs used was Nvidia Tesla T4 which using the cuda capability was able to train the neural network a lot faster than traditional CPUs. The input size of the sketch2image was 256x256x1 where 1 represents the image channel where-as for mask2Image the input was a one-hot-encoding feature representation of dimension 19, each representing a facial feature. The output block for sketch to image was a 3-channel Transpose Convolution block which generated a 3-channel image with a size of 256x256.

The total number of images in the training set was 15k and the validation set consists of 2k images. For the Mask2Image model, the training is done in steps that are equivalent to processing mini-batch gradient, so the mask-to-face model was trained for 30000 steps, which is equivalent to 30000 * 6 = 180000 /15000 images which is 12 epochs and the testing is done at intervals of 4000 steps. For the Sketch2Image model, the training was done for 24000 steps so 24000 * 6 = 144000 images processed which is equivalent to 144000 / 15000 = 10 epochs whereas the testing is done at intervals of 500 steps. Every 100 steps took around 8 minutes when calculated converting to around 4 hours per epoch. around 10 sec. The hyperparameters for this experiment were as follows

Model's Performance: The below images show the model's performance while training and testing based on the individual loss value with each step and a combined loss function which is the addition of individual loss functions mentioned in Regularization loss, LPIPS, and Pixel-wise L2 loss in the previous section. These plots are obtained through the tensorboard. However, the provided plots cover only 16,000 steps due to a Colab disconnection issue.

Explanation: The training losses for the map2face model exhibit a consistent downward trend, indicating ongoing learning and improvement. This suggests that the mapping process while dataset generation may differ from the paper, affecting performance. The generated faces become progres-

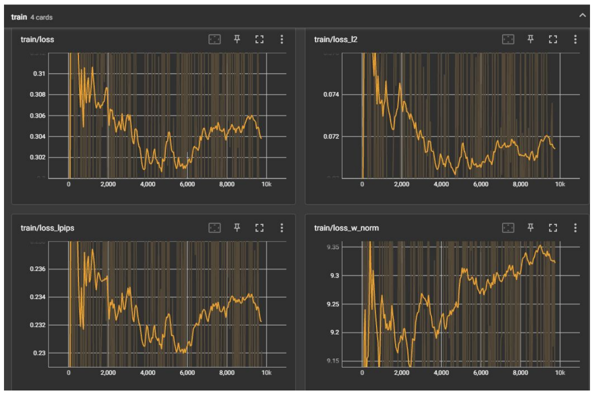
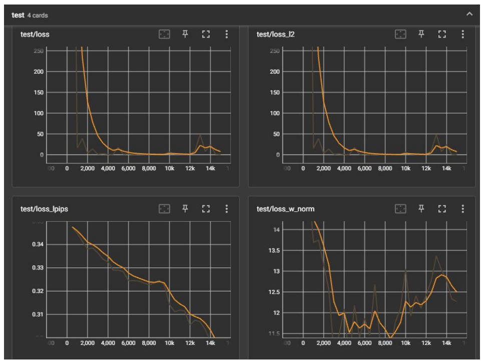
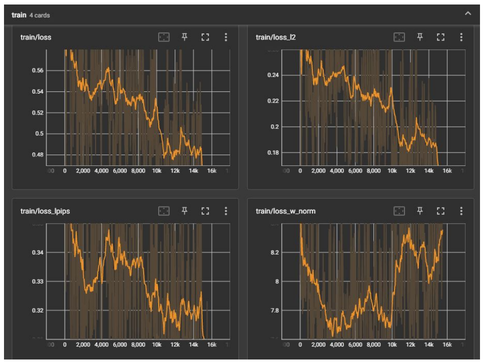


Figure 4: Loss graphs of Segmentation2Image in training

sively refined with clearer boundaries as training progresses. Training for more epochs is expected to yield even better results. Unfortunately, the Segmentation2Image experiment's validation loss graph is excluded due to a single-value log at 4000 steps



(a) Loss graph for Sketch2face in testing



(b) Loss graph for Sketch2face in training

Explanation: The below image (figure 5) shows the training and testing losses of the model for the sketch-to-face model. The training and testing losses for the sketch-to-face model show a clear downward trend, indicating ongoing learning and improvement. This suggests that the model is effectively adapting to the dataset, potentially due to the image-processing steps employed during dataset generation. The smooth pattern of the testing total loss curve further supports the model's ability to adjust its weights to accommodate variations in data distribution. More comprehensive plots in the attached code base can provide further insights into the model's performance and learning process. Continued training is expected to yield even better results, enhancing the model's ability to generate accurate and refined facial images.

5. Results

Figure 4 shows the results we obtained from the implementation we have done.

Sketch: Even though our model has undergone different data processing compared to the model described in the paper, we have observed that our model can still generalize well to their sketches, yielding comparable results to those in the paper. It is worth noting that our implementation produces more diverse outputs that retain finer details, such as facial hair, which are also retained in the paper.

Segmentation: We have also observed similar results in the image generation from segmentation tasks, where our implementation can generate more detailed and clear images, which is consistent with the results presented in the paper.

Generally, for both models: Upon analyzing the learning curves, we observed a reduction in the training loss over iterations. However, an interesting observation was made during the testing phase:

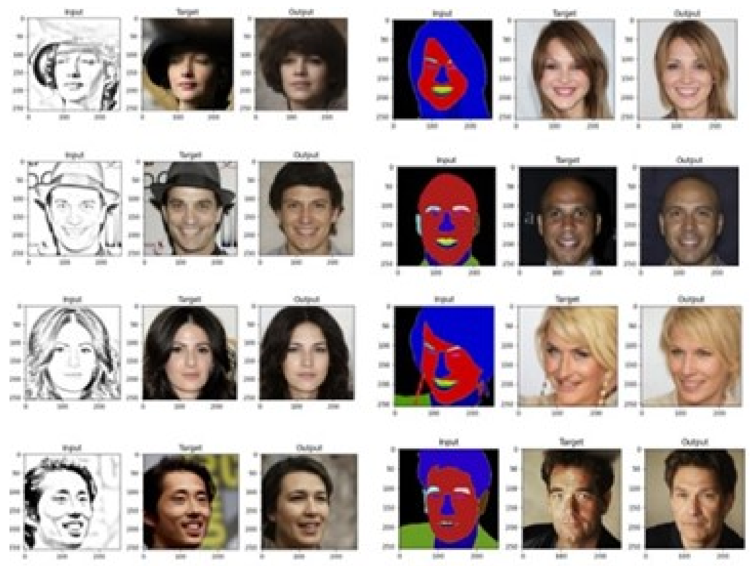


Figure 6: Results

the model produced good results on the test data despite the loss not updating significantly. We hypothesized that this could be attributed to the effect of the pretrained checkpoint from which the model training was initialized. The pretrained weights might have provided a strong foundation for the model to generate satisfactory results even without significant updates to the loss. In terms of feature recognition, our model demonstrated impressive performance. It successfully recognized and referenced various facial features such as hairstyle, nose, lips, jaw size, beard, and the orientation or angle of the faces. This indicates that the model learned to capture and reproduce these characteristics accurately, contributing to the realism of the generated images. Furthermore, there was a noticeable improvement in the prediction of eye features. Although the exact extent of improvement may vary, the model showed promising progress in generating realistic and well-defined eyes. This is a crucial aspect of facial synthesis, as the eyes play a significant role in determining the overall appearance and expression of a face, especially in the sketch2face model.

Overall, these findings highlight the effectiveness of the model in capturing and reproducing important facial features. Despite the loss not updating significantly during training, the model demonstrated satisfactory performance on the test data. These results indicate the successful utilization of the pretrained checkpoint and the ability of the model to generate high-quality and visually appealing facial images.

6. Conclusion:

Our project was aimed at gaining a thorough understanding of Generative Adversarial Networks (GANs) and their potential for picture production. Specifically, we focused on the Pixel2style2pixel (pSp) framework, a cutting-edge approach for picture synthesis, and evaluated its performance in comparison to other approaches.

After a detailed analysis of the pSp framework and other existing methods, we set up an experimental environment using PyTorch and other relevant libraries to conduct our testing. We constructed the Pixel2style2pixel (pSp) framework and a StyleGAN encoder and created our own sketches and maps to generate the dataset from the CelebA-HQ dataset.

After setting up our experimental environment and implementing the pSp framework, we trained the model using the CelebAMask-HQ dataset. We evaluated the pSp framework's effectiveness using the datasets and examined its ability to handle a variety of inputs, including a range of facial expressions. We also generated a loss graph from the logs obtained during the training process.

Our project has contributed to the field of computer vision and enhanced the state-of-the-art in picture generation by investigating the pSp framework and evaluating its performance in comparison to other methods. We have gained valuable insights into GANs and their potential for picture production, and we look forward to further exploring this exciting field.

7. Appendix:

Dataset Description:

A dataset called **CelebAMask-HQ**[15] is used for training the model and it consists of 30,000 high-resolution face images that have been manually annotated with segmentation masks of facial attributes. These attributes include all of the facial features and accessories, such as the skin, nose, eyes, eyebrows, ears, mouth, lip, hair, hat, eyeglass, earring, necklace, neck, and clothing. The CelebA-HQ methodology is used to create the dataset, which is generated from the CelebA dataset. CelebAMask-HQ is a useful tool for testing and refining algorithms in various fields because of its fine-grained mask annotations.

Future work

As a future work, we plan to explore the performance of PSP net compared to Pix2PixHD, which we were not able to do in this paper due to limited compute and resources. This would allow us to evaluate the effectiveness of PSP net in image-to-image translation tasks and further improve its capabilities.

Additionally, we plan to investigate the use of different feature extractor models in PSP net. While we used ResNet in this paper, we believe that exploring other architectures such as Inception could improve the performance of the network and allow us to tackle a wider range of image translation tasks. This could lead to better results and more efficient use of resources.

In addition, we plan to evaluate and compare the performance of PSP net with CycleGAN as a part of future work. This would allow us to assess the strengths and weaknesses of each method in image-to-image translation tasks and provide insights on which approach is more suitable for different applications. Moreover, this would enable us to gain a deeper understanding of the limitations of PSP net and identify areas for improvement.

Problems encountered:

In order to incorporate a new activation function into the model, we encountered a challenge as the activation function was not available in the PyTorch library. As a solution, we decided to implement our own PyTorch operation. However, this required the Ninja C++ compiler, which we encountered difficulties installing on Windows. To overcome this obstacle, we attempted to use a virtual machine (VM) but encountered another issue where the GPU was not accessible within the VM. As a result, we opted to use Google Colab, specifically the pro version, to compile the FusedLeakyReLU activation function and train the models. However, we faced additional challenges related to the proper updating of images in Google Drive and limited compute hours. Consequently, we had to work with a subset of the dataset, consisting of 15,000 images, to accommodate the resource constraints and ensure the successful execution of the training process.

Due to the project's complexity and the extensive training time required for pix2Style2pix, we were unable to fully explore the practical implementation of pix2pixHD. We had discussions with the professor regarding this matter, and he advised us to provide a detailed explanation of the other comparative studies instead. The evaluation process for pix2pixHD was time-consuming and demanded significant effort, which we were unable to accommodate within the project's constraints.

Code:

For inference of the model refer to this code:
https://colab.research.google.com/drive/1bxqslyfFmDMzVSZ9ftRPG8LMlM85QAaX?usp=sharing

For training refer:
https://drive.google.com/file/d/1tbs7pVGcjT5gG_CCFwJIF_Y0y3vlMrA_/view?usp=share_link

For result images and model checkpoints refer:
https://drive.google.com/drive/folders/1pzW1d7Tfjd8xh6N8zFLn7A9nlBNHq0yo?usp=share_link

References

[1] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, Daniel Cohen-Or. Encoding in Style: a StyleGAN Encoder for Image-to-Image Translation. arXiv, 2021.

[2] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 44014410, 2019.

[3] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 81108119, 2020.

[4] Zhu, J., Krähenbühl, P., Shechtman, E., and Efros, A.A. (2016). Generative Visual Manipulation on the Natural Image Manifold. European Conference on Computer Vision.

[5] Esther Robb, Wen-Sheng Chu, Abhishek Kumar, and Jia- Bin Huang. Few-shot adaptation of generative adversarial networks. arXiv, 2020.

[6] Antonia Creswell and Anil Anthony Bharath. Inverting the generator of a generative adversarial network. IEEE transactions on neural networks and learning systems, 30(7):1967 1974, 2018.

[7] Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto. Adversarial latent autoencoders. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1410414113, 2020.

[8] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pt'erez, Michael Zollhĺofer, and Christian Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images. arXiv preprint arXiv:2004.00121, 2020.

[9] Ali Jahanian, Lucy Chai, and Phillip Isola. On the steerability of generative adversarial networks. arXiv preprint arXiv:1907.07171, 2019.

[10] Edo Collins, Raja Bala, Bob Price, and Sabine Susstrunk. Editing in style: Uncovering the local semantics of gans. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 57715780, 2020.

[11] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 11251134, 2017.

[12] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 87988807, 2018.

[13] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 81888197, 2020.

[14] Pidhorskyi, S., Adjeroh, D.A., and Doretto, G. (2020). Adversarial Latent Autoencoders. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 14092-14101.

[15] Zhu, J., Shen, Y., Zhao, D., and Zhou, B. (2020). In-Domain GAN Inversion for Real Image Editing. ArXiv, abs/2004.00049.

[16] Lee, C., Liu, Z., Wu, L., and Luo, P. (2019). MaskGAN: Towards Diverse and Interactive Facial Image Manipulation. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 5548-5557.