Module: INM705 - Deep Learning I
Professor: Alex Ter-Sarkisov

PHOTO-TO-ANIME: IMAGE TRANSLATION WITH GENERATIVE ADVERSARIAL NETWORKS

Authors:
Elisabeta Monica Furdui - 190045971
Cansin Aysegul Sapmaz - 190045685

1.Introduction

Image-to-image translation is part of a class of vision problems where the task is to learn the mapping between images from one domain to images from a different domain and transform the source images so they have the style as the images from the target domain.
This project is based on the research paper "U-GAT-IT: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation" (Kim et al., 2020) which proposes U-GAT-IT, a new method of unsupervised image-to-image translation based on a novel attention module and learnable normalization function.
Our project attempts to improve on the results of the paper (Kim et al., 2020) by adding a facial keypoint predictor model to the ensemble, and using the predicted keypoint coordinates to make a new loss function. We experiment with a new dataset that contains facial keypoint annotations. The baseline model, U-GAT-IT, uses one generator to translate source images into anime style, and then a different generator to translate the anime style output back to its original form (Kim et al., 2020). This process is used by Kim et al. to prevent mode collapse by making sure the anime style output encodes sufficient information about the source image (2020). While training the original U-GAT-IT model, we have observed that the quality of reconstructed images was highly correlated with the quality of anime style translations. Inspired by this observation, we use our new keypoint predictor on the reconstructed source images. We then train the generators to minimize the difference between keypoint coordinates of the reconstructed images and the original source images.

2.Methodology

2.1.Dataset Details

The dataset for this project is composed of two parts: the selfie dataset and the anime dataset. The research paper (Kim et al., 2020) uses 3400 female selfie images of size 256x256 for training and another 100 images for testing. The anime dataset has 3400 female anime images for training and 100 for testing (Kim et al., 2020).
We discarded the selfie dataset of the paper as it does not have any attribute annotations. We used CelebFaces Attributes Dataset (Liu et al., 2015) instead to sample 3405 images for training and 100 images for testing. We resized the images to equal height and width and adjusted the padding variable accordingly in the code. The anime dataset is the same dataset as in the paper, only resized to match the size of CelebFaces Attributes Dataset (Kim et al., 2020).
For processing the images, the same data transformation of the baseline model is used; all images are normalized to have a mean and standard deviation of 0.5 in all three channels. The baseline model performs random cropping and mirroring on the training dataset to simulate a larger training set. Due to having to keep the facial keypoint locations undisturbed, we disabled this feature.

The attribute annotation file (Liu et al., 2015) provides the name of the image file together with x and y coordinates of 10 attribute points in this order : left eye x-axis coordinate , left eye y-axis coordinate, right eye x-axis coordinate, right eye y-axis coordinate , nose x-axis coordinate, nose y-axis coordinate, left mouth x-axis coordinate, left mouth y-axis coordinate, right mouth x-axis coordinate, right mouth y-axis coordinate. These coordinates are read into a dictionary for processing. The number of pixels cropped for resizing the photos is subtracted from the y-axis coordinates.

2.2.Baseline Model

The model of the paper (Kim et al., 2020) which we are using as a baseline consists of two generators Gs→t(source to target) and Gt→s(target to source) and two discriminators Ds(source) and Dt(target). As seen in Figure 1, the attention module is in both the generator and the discriminator. The generator consists of an encoder, a decoder and an auxiliary classifier, while the discriminator is composed of an encoder, a classifier and an auxiliary classifier (Kim et al., 2020).

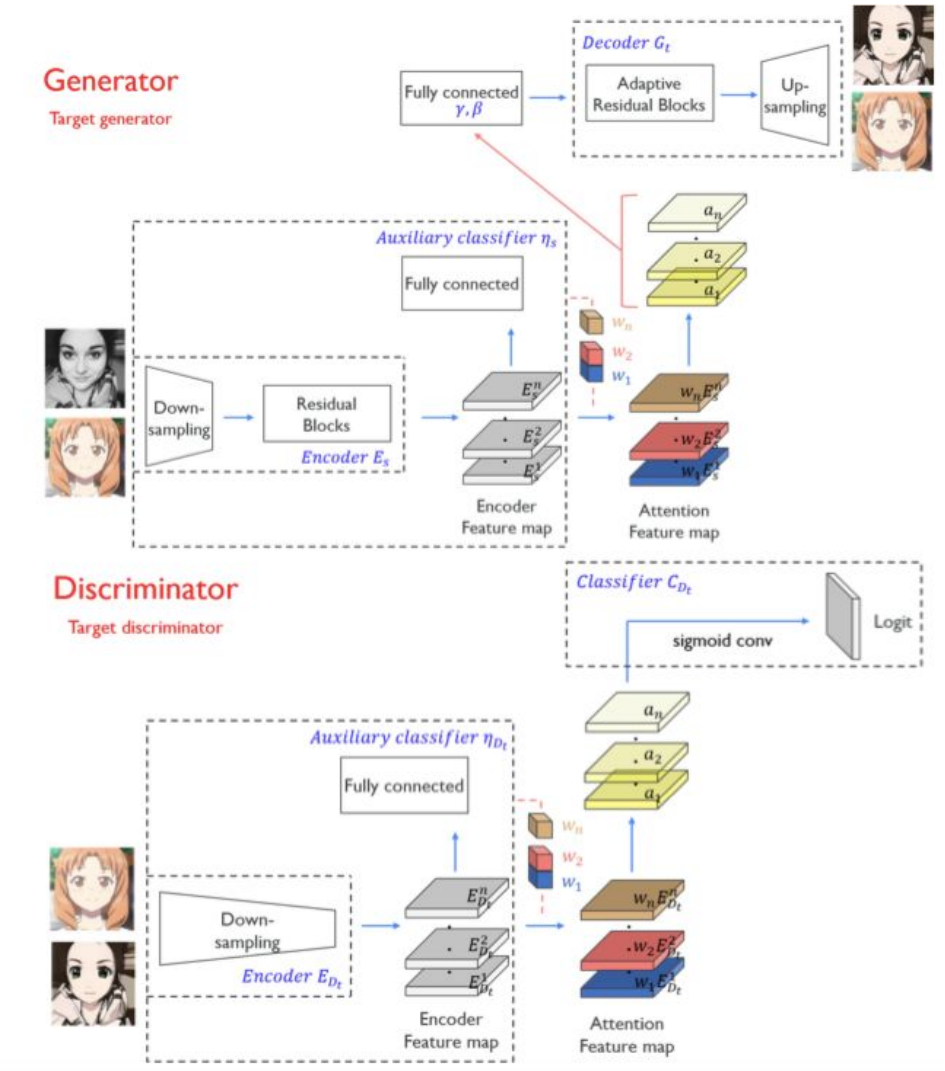


Figure1.Architecture of U-GAT-IT (Kim et al., 2020)

The model is trained on a combination of four different loss functions. The first function is adversarial loss, which lets the generator's distribution converge to target distribution (Kim et al., 2020). This is the traditional loss criterion of GANs that let them improve generator results based on discriminator outputs (Goodfellow et al., 2014). The second loss function is called cycle loss; this is the solution Kim et al. propose to handle mode collapse (2020). This criterion is based on reconstruction of translated images. Source human photos are first translated into anime images with the source-to-target generator, then the resulting anime images are translated back to source photos with the target-to source generator (Kim et al., 2020). Normalized, pixel-wise loss between the original photo and the reconstructed image is minimized during training. This reconstruction process is applied both to human and anime images. The third loss function is identity loss; pixel-wise difference between the image and its translated output from the generator is calculated to minimize colour differences during style transfer (Kim et al., 2020). Finally, Class Activation Map (CAM) loss uses the attention maps that are produced by the auxiliary discriminators to decide which regions to improve during generation (Kim et al., 2020). The model is trained in "iterations" instead of epochs. Each iteration is the model being trained on a single image in the training set. One epoch is thus 3405 iterations of training.

2.3.Our Experiments

As explained in section 2.2, the baseline model translates source images to target style and then reconstructs the translated image back to its original self using a second generator. Then, the loss between reconstructed image and the original image is used to optimize the model. Thus, the quality of reconstruction has a direct effect on source-to-target translation. Inspired by this mechanism, we have decided to work on improving reconstruction of source images. One problem that we have observed in the baseline model was that the translated images would not keep their original facial expressions. To solve this problem, we added a facial keypoint predictor to the model, and trained both the source-to-target and the target-to-source generator to minimize the difference between keypoint locations of the original image and the reconstructed image.

2.3.1.Keypoint Predictor

The keypoint predictor was implemented using the official Pytorch documentation (pytorch.org, n.d.) for training an image classifier. The network architecture proposed in the tutorial was slightly modified and trained using a different loss function to perform regression of facial keypoint coordinates. The network architecture and training specifications are given in figure 2.

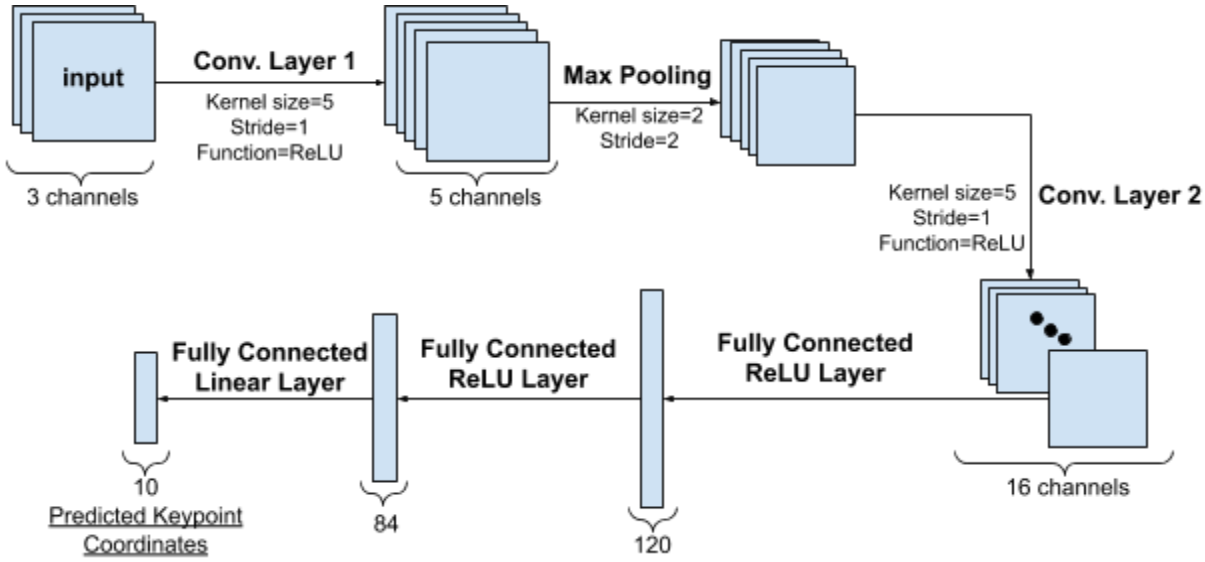


Figure 2: Architecture and specifications of the keypoint predictor.

2.3.2. Ensembling the Baseline Model and the Keypoint Predictor

We used the keypoint predictor on the reconstructed celebrity photos. Photos are first translated into anime using source-to-target generator, and then they are translated back to celebrity photos using target-to-source generator by the baseline model. The keypoint predictor then takes these reconstructions and regresses their keypoint locations that include left eye, right eye, nose, and mouth edges. Taking the original celebrity photos' keypoint coordinates as the target output, mean squared loss between target and predicted values is calculated. We then tried several approaches to combine this keypoint loss with the other four loss functions of the baseline model.

Method 1 was to add the keypoint loss to the weighted sum of the four loss functions, multiplied by cycle weight. Cycle weight is the weight applied to cycle loss, which is the pixel-wise loss between the original and the reconstructed image. The default value of cycle weight is chosen to be 10 in the baseline model. We tried this method in particular because both cycle loss and keypoint loss aimed to improve reconstruction, so grouping both losses in the same category and weighting them equally for optimization was intuitive.

Method 2 was to add the keypoint loss to the sum multiplied by adversarial weight. This is the weight applied to adversarial loss in the baseline model. This choice was motivated by the observation that keypoint loss and adversarial loss were numerically in similar ranges after being trained for roughly 3 epochs, so our addition would not overpower adversarial loss while having the same weight. Adversarial weight is chosen to be 1 in the baseline model.

In both of these methods, once the keypoint loss was added, the model optimized both the source-to-target and target-to-source generators to minimize the weighted sum of losses.
We tried another approach, method 3, where we did not add the keypoint loss with the other loss functions, but used it separately to optimize the source-to-target generator only. In this approach, both source-to-target and target-to-source generators were optimized to minimize the weighted sum of the four loss functions just as they were in the baseline model. Then, we used a separate Adam optimizer and optimized the source-to-target generator one more time to minimize the keypoint loss only.

3.Results

3.1. Different Approaches to Using the Keypoint Loss

Among the three methods we tried to combine keypoint loss with the other loss functions, method 2, multiplying the keypoint loss by adversarial weight and adding it to the other four loss functions' weighted sums, gave us the best results. Furthermore, we found that how much the keypoint model was trained beforehand had a strong effect on results. Training the keypoint model until it converged before starting the GAN's training stopped the model from translating the photos into anime. We believe this to be a similar effect to overtraining the discriminator beforehand, and thus not letting the generators catch up. To combat this issue, we tried matching the keypoint model's training loss to the discriminator loss and received higher quality results. The learning curves of the keypoint predictor and the discriminator are not the same, so we had to use different epochs for the models. The discriminator loss and the keypoint model's training loss during the first 10K iterations and the first 3 epochs respectively can be seen in figure 3 as a comparison. We had the best results by dividing the GAN training into 10K-iteration batches, which is roughly 3 epochs with our dataset of 3405 images. Before the first 10K, we trained the keypoint model 3 epochs. Then before the second 10K, we train the keypoint model 20 more epochs. Before the third 10K, the keypoint model is trained 40 more epochs. Lastly, before the fourth 10K, the keypoint model is trained 20 more epochs. After 80 epochs the keypoint model converges, so it is not trained after that point.
Method 1, multiplying the keypoint loss by cycle weight gave us anime style results, however the visual quality improvement over iteration was extremely slow so this approach was not explored further.
Method 3, not adding the keypoint loss to the weighted sum of other loss functions, and optimizing the source-to-target generator twice to minimize it, resulted in images that were not translated into anime. We believe the keypoint loss simply overpowered the other loss functions, and the second optimization overwrote the updates of the first optimization. The results of different approaches we tried can be seen and compared in Figure 4.

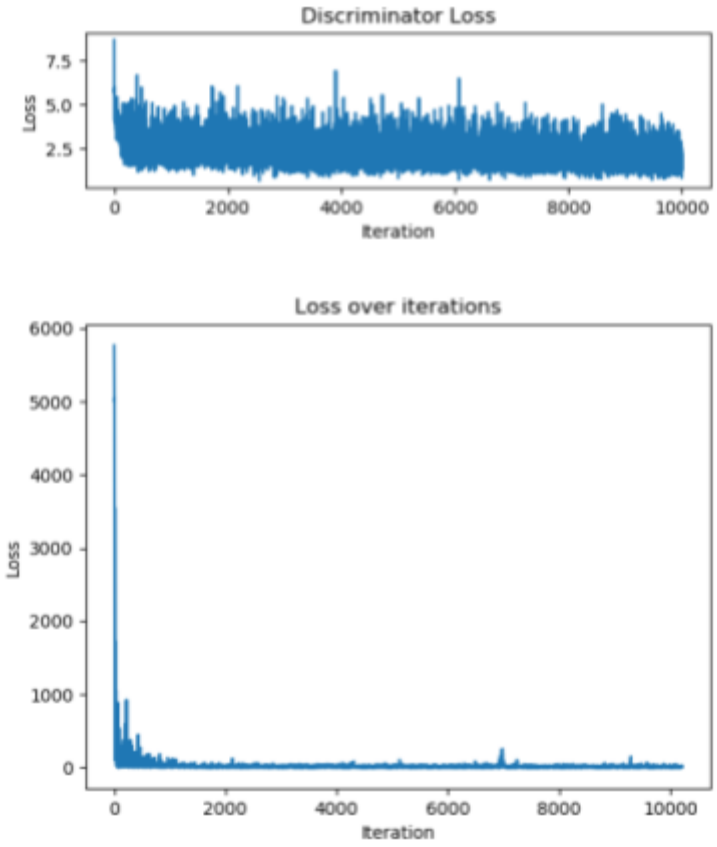


Figure 3: Discriminator loss (above) and keypoint model training loss (below). After 10K iterations, the keypoint model can predict coordinates with an average loss of 2.5.

The overpowering effect of keypoint loss in method 3 can also be seen in the heat maps shown in Figure 4.a. Mouth, nose, and eyes are focused more by the model compared to the results of method 2 shown in figure 4.b and 4.c.

For comparing our results with the baseline, we used method 2 with the training specifications that gave us the results in figure 4.c as it showed the best performance.

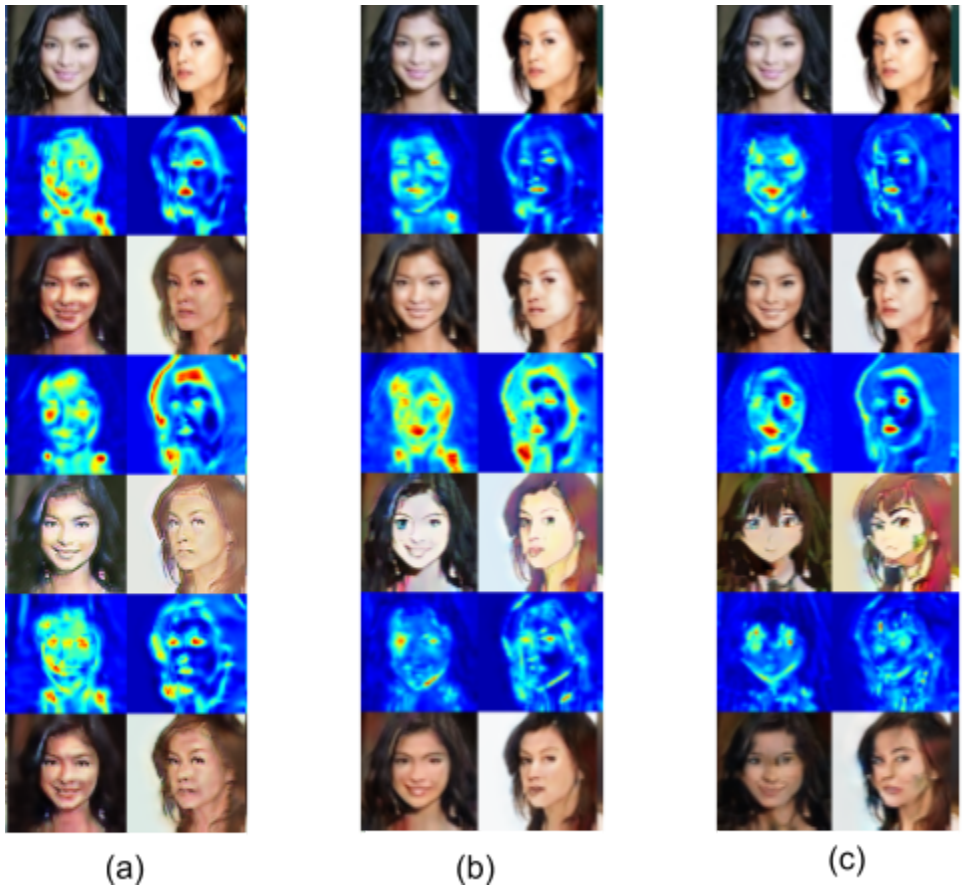
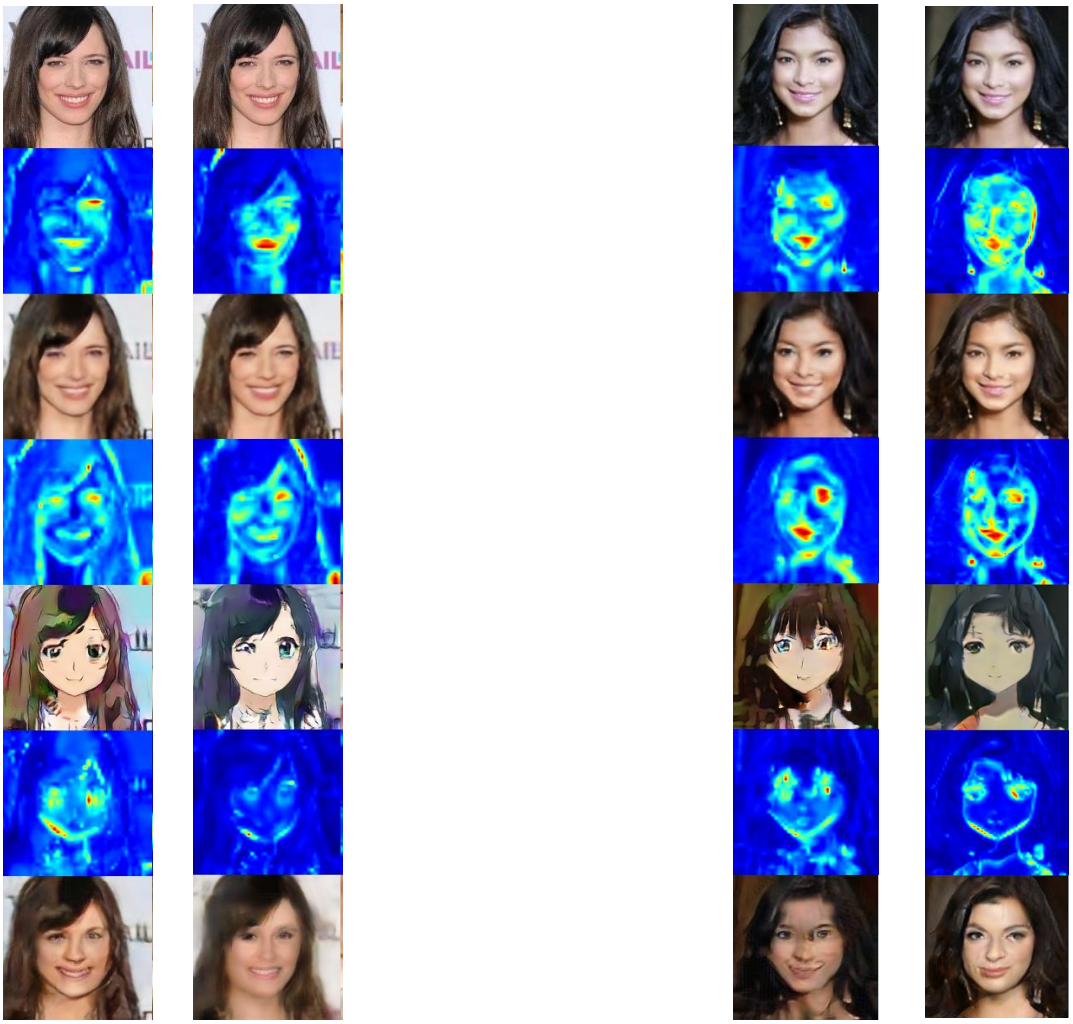


Figure 4: 40K iteration results of method 3 (optimizing the source-to-target network twice) (a) and method 2 (adding keypoint loss to other losses with adversarial weight) (b and c). In (b), the keypoint model was trained until convergence beforehand. In (c), the keypoint model was trained to match the discriminator's performance as explained in section 3.1.

3.2. Visual Quality

As seen in Figure 5 and particularly in Figure 6 the keypoint model anime is preserving the facial expression better than the baseline model. The smile is clearly defined and the eyes have a round shape in both 40k iterations and 50k iterations of the generated anime. The baseline model seems to change the hair color of the anime to dark and lighten the skin color of the face, however the keypoint version of the anime is able to preserve the skin and hair color of the original image which is especially noticeable at the 50k interactions sample image. At the same time, in the keypoint model, the heatmap is focusing on the relevant details of the face such as the eyes, nose and mouth in comparison to the baseline.



Keypoint Baseline Keypoint Baseline

(a) 50k iterations (b) 40k iterations

Figure 5. Visual quality of sample generated images of keypoint and baseline models.

Figure 6. Visual quality of sample generated image at 50k iterations of keypoint and baseline models.

3.3. Frechet Inception Distance (FID)

The authors of the paper "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium " (Heusel et al., 2018) proposed Frechet Inception Distance (FID) as a measure of the quality for images specifically generated by GANs. FID uses the Inception V3 model to capture features and compares statistically real images with synthetic images using mean and covariance. A lower FID score means a better similarity to the original and a better synthetic image.

In Figure 8, we have a one by one comparison of the human-to-reconstructed images. Visually we can see that keypoint reconstructed images are better, which is confirmed by the FID score as well. The keypoint reconstructed images obtained a better score in both sampled images.
We also used FID to compare dataset results as seen in Figure 9. The human-to-reconstructed image of the keypoint model obtained a better FID score than the baseline, the human-to-anime and anime-to-human images performed slightly worse.

Model	Human to Reconstructed	Human to Anime	Anime to Human
Baseline	**149.25**	**133.49**	**137.68**
Keypoint	**121.16**	**144**	**140.07**

Figure 7. FID comparisons using the test datasets, at 50k iterations.

(a) Original image (b) Keypoint model (c) Baseline model

Figure 8. Sample image results from 50k iterations (top row) and 40k iterations (bottom row) used for FID comparison.

Keypoint model	Baseline model
93.92	170.48
120.90	171.85

Figure 9. FID results at 50k iterations (top row) and 40k iterations (bottom row)

4. Discussion

In this project, we added a facial keypoint predictor and a new loss function to the U-GAT-IT model from the paper “U-GAT-IT: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation” (Kim et al., 2020). In order to

improve the reconstructed images of the baseline model a facial keypoint predictor was added to the model, and trained both for the source-to-target and the target-to-source generator in order to minimize the difference between keypoint locations of the original image and the reconstructed image. We then conducted experiments to combine the keypoint loss with the other four loss functions of the baseline model. Multiplying the keypoint loss by adversarial weight and adding it to the other four loss functions' weighted sums gave us the best results. The keypoint model was trained to match the discriminator's performance.

In terms of the results, the visual quality of the anime images using our model was improved in regards to expressivity (eye shape, nose, smile) and color (skin and hair). The heatmaps show an increased attention on the relevant features of the face such as eyes, nose and mouth.

We used Frechet Inception Distance to compare our human-to-reconstructed, human-to-anime and anime-to-human images with the baseline results. The human-to-reconstructed image dataset of the keypoint model obtained a better FID score than the baseline, the human-to-anime and anime-to-human images performed slightly worse. We also did a one by one comparison of the human-to-reconstructed images from keypoint and baseline. As seen in Figure 8. and Figure 9. , keypoint reconstructed images obtained a better score in comparison to baseline in both the sample images.

5. Future work

Convolutional neural networks discard features due to max-pooling and they lack spatial relationship among features, as such, we propose as future work the use of Capsule Networks. Capsule Networks were first introduced by the paper titled "Dynamic Routing Between Capsules" (Sabour, Frosst and Hinton, 2017). Capsule networks are based on the idea of inverse graphics, you start with an image and you find what objects it contains and their instantiation parameters such as orientation and so forth. The features are represented in vector form and the length of the output vector is the probability of detection of feature. The capsule weights are learned through dynamic routing.

Using the idea of CapsuleGan (Jaiswal et al., 2018) we can replace the discriminators of U-GAT-IT with capsule networks. The paper "CapsuleGAN: Generative Adversarial Capsule Network" (Jaiswal et al., 2018) conducted experiments on MNIST and CIFAR-10 datasets and demonstrated that CapsuleGans outperformed GANs based on CNNs. It would be interesting to see how CapsuleGan (Jaiswal et al., 2018) performs on more complex images such as the datasets used in this project and if better results would be obtained.

6. References

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014). *Generative Adversarial Nets*. [online] Available at: https://arxiv.org/pdf/1406.2661.pdf.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B. and Hochreiter, S. (2018). *GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium*. [online] Available at: https://arxiv.org/pdf/1706.08500.pdf [Accessed 1 May 2020].

Jaiswal, A., Abdalmageed, W., Wu, Y. and Natarajan, P. (2018). *CapsuleGAN: Generative Adversarial Capsule Network*. [online] Available at: https://arxiv.org/pdf/1802.06167.pdf [Accessed 24 May 2020].

Kim, J., Kim, M., Kang, H. and Lee, K. (2020). *U-GAT-IT: UNSUPERVISED GENERATIVE ATTEN- TIONAL NETWORKS WITH ADAPTIVE LAYER- INSTANCE NORMALIZATION FOR IMAGE-TO-IMAGE TRANSLATION*. [online] Available at: https://arxiv.org/pdf/1907.10830.pdf [Accessed 11 May 2020].

Liu, Z., Luo, P., Wang, X. and Tang, X. (2015). *Deep Learning Face Attributes in the Wild **. [online] Available at: https://arxiv.org/pdf/1411.7766.pdf [Accessed 22 May 2020].

pytorch.org. (n.d.). *Training a Classifier — PyTorch Tutorials 1.5.0 documentation*. [online] Available at: https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html [Accessed 30 May 2020].

Sabour, S., Frosst, N. and Hinton, G. (2017). *Dynamic Routing Between Capsules*. [online] Available at: https://arxiv.org/pdf/1710.09829.pdf [Accessed 7 Mar. 2020].