IE643: Deep Learning - Theory and Practice **July-Nov 2022**

Enabling Deep Hierarchical I2I Translation by Transferring from GANs

Team Name: Zero1 *Team Members: 22D1594*

Abstract

Training of GANs can be an extremely difficult process. Even with fine-tuned hyper-parameters and previously trained result, we can have mode collapse after retraining. In this work we have explored a popular architecture for performing automated image to image translation, which can be used widely in the computer graphics industry, for automatic creation of morphing and other on-demand tasks, that requires expertise and human resources. We have built a loss function, which performs slightly better than the previously reported results when trained from scratch. The architecture is applied on a new dataset and we have some good as well as failure cases for this study.

1 Introduction

Image to Image *(I2I)* translation is used in the field of Computer Graphics (CG), especially in the movie industries. Traditionally this is a labor intensive process, and the proposed technique can be applied to automatic translation of faces/objects. Researchers are trying out various deep learning techniques for this task because deep learning techniques provide neat latent transitions by projecting the latent vector to images of our choice. The common architecture that is used for synthesis of images is GANs. Here we use this technique for translation which not only gives consistent images but also generates high resolution images when translating it from one class to another. I2I translations show inferior performance when translations between classes require large shape changes, for example when translating images from ant to elephant etc. in current deep learning architectures.

This is the work done by Yaxing Wang, Lu Yu and Joost van de Weijer, which was presented at the NeurIPS 2020 conference. The main objective of the model is to learn the representations by leveraging hierarchical features, which contains:

- Structural information in the shallow layers.
- Semantic information extracted from the deep layers.

They have implemented a novel transfer learning method by transferring knowledge from pre-trained GANs, enabling learning on small datasets. They leveraged the discriminator of pretrained GAN to initialize the encoder and discriminator, and leveraged the pretrained generator to initialize the generator of their model. They have also introduced Adaptor networks to address the alignment problem between encoder and decoder when using knowledge transfer. They are first to do I2I translation over 1000 classes in animal faces and food datasets. They qualitatively and quantitatively showed that transfer learning significantly improves the performance of I2I systems for small datasets.

The main contribution of our projects are:

- We have been successful in replicating the results of the original author's pipeline.

- We have used a different dataset, named as NABirds dataset [1] , for which we have done the evaluation studies.

- We have designed different loss functions such as SSIM [12] and Softplus loss for GAN based training, in which we have shown that our Softplus loss performs marginally better than their original loss.

The report is structured as follows, we provide a survey of existing literature in Section 2. Our proposal for the project is described in Section 3. We give details on experiments in Section 5. A description of future work is given in Section 7. We conclude with a short summary and pointers to forthcoming work in Section 8.

2 Literature Survey

Generative Adversarial Networks [3] or GANs are used to generate realistic samples by using an input distribution by playing a min max game with the discriminator and generator. Here, we will focus on only those GANs which are used for image to image translations. Pix2Pix [5] GAN was the early version of I2I translation models where the researchers examined conditional GANs as the general purpose solution for I2I translation problems. This model can be used to perform a wide variety of tasks, including effectively synthesize photos from label maps, reconstructing objects from edge maps and colourising images among other tasks. CycleGAN [13] is used to translate images from a source domain X to a target domain Y in the absence of any paired examples. The researchers have introduced a cycle consistency loss to check whether the translated images are consistent with the source images and vice versa. This model can achieve several tasks like style transfer using different collections, object transfiguration, season transfer etc. StarGAN [2] uses a scalable approach that performs I2I translations for multiple domains using a single model. It does simultaneous training of multiple domains with a single network. It produced high visual quality compared to other methods at that time. The Generator G takes both image and target domain label and generates fake label. It tries to reconstruct the original image from the fake image by using the original domain label.

BigGAN [1] was the pioneer model used to generate high resolution diverse complex samples from the ImageNet dataset. Previous architectures were not good at generating samples when the images were scaled up, and from this model onwards, GANs were used to generate high resolution images. The architecture used Class-conditional image synthesis, which obtained a good Inception Score and Frechet Inception distance for most of the generated images. This model can also be used for I2I translation between high resolution images. SDIT [10] was the first model to perform diverse domain translation using a single generator. Previous models used different generators when the style of the content changes etc. This method also uses attention to focus the Generator on specific attributes.

Researchers of StyleGAN [6] borrowed ideas from style transfer literature to apply in the Generator of GANs. The architecture automatically learns unsupervised separation of high level attributes, e.g. pose, identity etc when trained on human faces. They build a method for synthesizing stochastic variation in generated images e.g., freckles, hair etc. that enables scale specific control synthesis. Other archjtectures like DRIT++ [7], MineGAN [9], and DMIT [11] are used for multi-modal unsupervised image to image translation [4].

[1] `https://dl.allaboutbirds.org/nabirds`

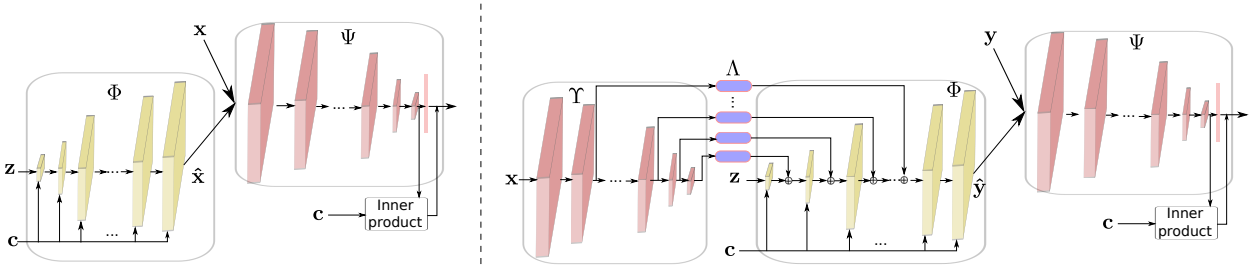


Figure 1: *Left*: the traditional form of conditional GAN (i.e., BigGAN) which contains the generator Φ and the discriminator Ψ. *Right*: the proposed DeepI2I method based on conditional GAN (left). The method consists of four terms: the encoder Υ, the adaptor Λ, the generator Φ and the discriminator Ψ. The encoder Υ is initialized by pre-trained discriminator (left), as well as both the generator Φ and the discriminator Ψ by pre-trained GANs (left). The adaptor Λ aims to align the pre-trained encoder Υ and the pre-trained generator Ψ.

3 Methods and Approaches

The main challenge lies in inverting deep low-resolution bottleneck (latent) representation into a high-fidelity image, since the deep layers contains many attribute level information from which it is difficult to reconstruct realistic images which closely follow the sturcture of input images. The proposed network used BigGAN architecture at its core, the encoder network follows the same architecture as the BigGAN discriminator. Novelties in I2I domain from this work include *orthogonal regularization* to control trade-off between variety and image quality, *class conditioning* vai a class embedding network which works at various depth in the network. This allows the network to translate between many class domains, in which the current architectures doesnot perform well.

Let $\mathcal{X}, \mathcal{Y} = \mathbb{R}^{H \times W \times 3}$ be the source and target domains. The architecture is composed of four neural networks: encoder Υ, adaptor Λ, generator Φ and discriminator Ψ. The architecture is shown in Figure 1 [2] They aim to learn a network to map the input source image $\mathbf{x} \in \mathcal{X}$ into a target domain image $\hat{\mathbf{y}} \in \mathcal{Y}$ conditioned on the target domain label $\mathbf{c} \in \{1, \ldots, C\}$ and a random noise vector $\mathbf{z} \in \mathbb{R}^{\mathbf{Z}}$, $\Phi\left(\Lambda\left(\Upsilon\left(\mathbf{x}\right)\right), \mathbf{c}, \mathbf{z}\right) \rightarrow \hat{\mathbf{y}} \in \mathcal{Y}$. Latent representation from different layers of encoder Υ is used to extract structural information (shallow layers) and semantic information (deep layers). Let $\Upsilon_l\left(x\right)$ be the l-th $(l = m, ..., n (n > m))$ ResBlock [3] output of the encoder, which is fed into the corresponding adaptor Λ_l, from which it continues as input to the corresponding layer of the generator.

Input image $\mathbf{x}$ is taken as input, and the hierarchical representation $\Upsilon\left(\mathbf{x}\right) = \{\Upsilon\left(\mathbf{x}\right)_l\}$ of input image $\mathbf{x}$ is extracted. The adaptor Λ takes the output of Υ as input, that is $\Lambda\left(\Upsilon\left(\mathbf{x}\right)\right) = \{\Lambda_l\}$, where (Λ_l) is the output of each adaptor Λ_l which is further summed to the activations of the corresponding layer of the generator Φ. In some cases when we train the DeepI2I from scratch, the adaptor could be the identity function. The generator takes as input the output of adaptor $\Lambda\left(\Upsilon\left(\mathbf{x}\right)\right)$, the random noise $\mathbf{z}$ and the target label $\mathbf{c}$, we will discuss the adaptor network in the upcoming slides. The generator Φ outputs a $\hat{\mathbf{y}} = \Phi\left(\Lambda\left(\Upsilon\left(\mathbf{x}\right)\right), \mathbf{z}, \mathbf{c}\right)$ which is supposed to mimic the distribution of the target domain images with label $\mathbf{c}$. Sampling different $\mathbf{z}$ leads to diverse output results $\hat{\mathbf{y}}$.

The discriminator has three functions:

- The first one is to distinguish real target images from generated images.
- The second one is to guide the generator Φ to synthesize images which belong to the class indicated

[2] Image retrieved from `https://arxiv.org/abs/2011.05867` paper.
[3] The encoder consists of a series of ResBlock. After each ResBlock the feature resolution is half of the previous one.

by $\mathbf{c}$.

- The last one is to compute the reconstruction loss, which aims to preserve a similar pose in both input source image $\mathbf{x}$ and the output $\Phi\left(\Lambda\left(\Upsilon\left(\mathbf{x}\right)\right),\mathbf{z},\mathbf{c}\right)$.

The reconstruction is computed from the discriminator based on the l-th ResBlock Ψ, and referred to by $\{\Psi_l\left(y\right)\}$. The overall loss is a multi-task objective comprising of:

- *A conditional adversarial loss* - It optimizes the adverserial game between the generator and the discriminator, i.e., maximize Ψ while minimize $\{\Upsilon,\Lambda,\Phi\}$ to generate class specific images which correspond to label $\mathbf{c}$.
- *Reconstruction loss* - guarantees that the synthesized image $\hat{\mathbf{y}}=\Phi\left(\Lambda\left(\Upsilon\left(\mathbf{x}\right)\right),\mathbf{z},\mathbf{c}\right)$ preserve the same pose as the input image $\mathbf{x}$.

Conditional adversarial loss employing GANs

$$\mathcal{L}_{adv}=\mathbb{E}_{y\sim\mathcal{Y}}\left[\log\Psi\left(\mathbf{y},\mathbf{c}\right)\right]+\mathbb{E}_{\hat{\mathbf{x}}\sim\mathcal{X},\mathbf{z}\sim p(\mathbf{z}),\mathbf{c}\sim p(\mathbf{c})}\left[\log(1-\Psi\left(\Phi\left(\Lambda\left(\Upsilon\left(\mathbf{x}\right)\right),\mathbf{z},\mathbf{c}\right),\mathbf{c}\right)\right] \tag{1}$$

Here $\mathbf{p}\left(\mathbf{z}\right)$ follows the normal distribution , and $\mathbf{p}\left(\mathbf{c}\right)$ is the domain label distribution.

Final loss is optimized by mini-max game

$$\{\Upsilon,\Lambda,\Phi,\Psi\}=\arg\min_{\Upsilon,\Lambda,\Phi}\max_{\Psi}\mathcal{L}_{adv}. \tag{2}$$

Reconstruction Loss- based on set of activations extracted from multiple layers of discriminator Ψ :

$$\mathcal{L}_{rec}=\sum_{l}\alpha_l\left\|\Psi\left(\mathbf{x}\right)-\Psi\left(\hat{\mathbf{y}}\right)\right\|_1 \tag{3}$$

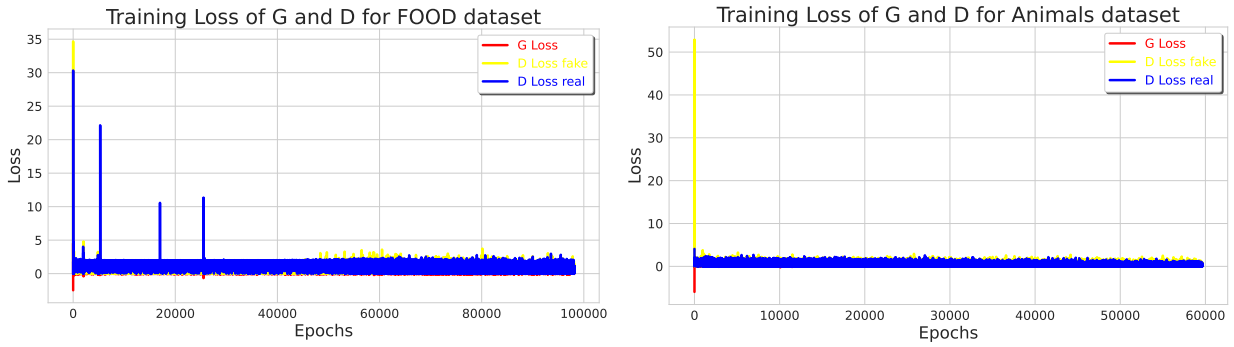
3.1 Knowledge Transfer

Knowledge transfer has not been used in I2I problems yet, which could help networks to learn from little labelled data. ImageNet is used as a universal knowledge transfer dataset, it is still unclear what can be used in I2I datasets. The authors argue that we could transfer the required knowledge from high quality pre-trained GANs to the architecture's encoder, generator and decoder. Pre-trained high-quality BigGAN is used to initialize the current architecture. Pre-trained discriminator Ψ is used to initialize discriminator Ψ of the proposed model as well as the encoder Υ, since, the discriminator of the BigGAN has the ability to correctly classify the input images on imageNet, which optimizes it to be an effective feature extractor. Pre-trained generator Φ is used to initialize the deepI2I generator Φ.

3.2 Adaptor Network

Transferring representations and abstractions of various resolutions between the encoder Υ and the generator Φ is done by connections at various layers by the adaptor network Λ.

$$\hat{\Phi}_l=\Phi_l+w_l\Lambda_l \tag{4}$$



(a) For food dataset

(b) For animal dataset

Figure 2: The loss generated for training the DeepI2I GAN using the Food and animal dataset.

Here, Φ_l is the output of the corresponding layer which has same resolution to Λ_l. Here $w_l = 0.1$, is used as hyper-parameters to balance the two terms. When we use the pre-trained encoder and the decoder network the representations are not aligned, for example the feature layer corresponding to the recognition of eyes in the encoder is not same as the feature layer which encourages the generation of eyes in the decoder. This is because there is no connection between the encoder and the decoder in BigGAN, and hence the adaptor network aligns the two representations.

3.3 Work done before mid-term project review

We have set up the initial training pipeline and trained the model for different datasets. The paper showed the experiment for Food and Animals dataset, and we have replicated the results. The graph obtained for training the food and animals dataset is shown in Figure 2. GAN training is extremely complicated and GANs are prone to mode collapse and stuff like that. We have used the number of iterations as mentioned in the paper and got the results.

Foods dataset - Number of parameters present in Generator = 70433988, Discriminator D = 87982370, Encoder = 87982370 and Adaptor = 87368065. The experiments took about 3 days to run for 98000 iterations which was trained from scratch. The training loss is shown in Figure 2a. Few samples are generated for the food dataset as shown in Figure 3. The translation for different classes for the food dataset is shown in Figure 4. This translation was done by taking the sampled latent vector and a target image, and translating the latent vector to the target image. Batch size of 4 was used for this dataset and the GAN was trained from scratch, a learning rate of 1e-04 was used for the Generator and a learning rate of 4e-04 was used for the discriminator. The results were satisfactory, given that there is a lot of issues while training GANs.

3.4 Work done after mid-term project review

NABirds (North America Birds) dataset - Number of parameters present in Generator = 70433988, Discriminator D = 87982370, Encoder = 87982370 and Adaptor = 87368065. The experiments took about 6 days to run for 151700 iterations which was trained from scratch. The training loss is shown in Figure 5a. Few samples are generated for the NABirds dataset as shown in Figure 7. The translation for different classes for the food dataset is shown in Figure 6. This translation was done by taking the sampled latent vector and a target image, and translating the latent vector to the target image. Batch size of 4 was used

(a) Few samples generated from food dataset.

(b) Few samples generated from food dataset.

Figure 3: Random samples are taken from the Generator and projected to similar latent space (Z=120 here).

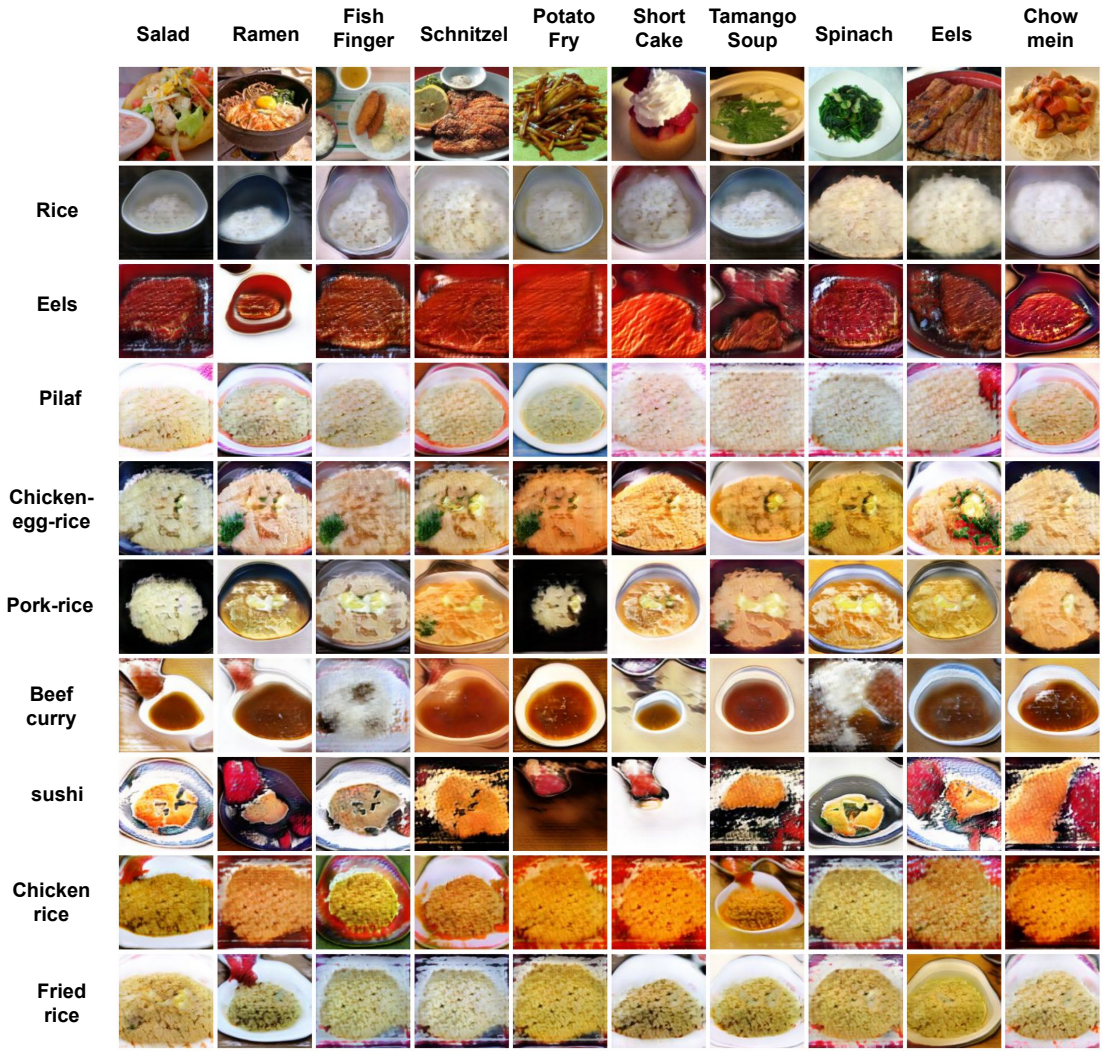
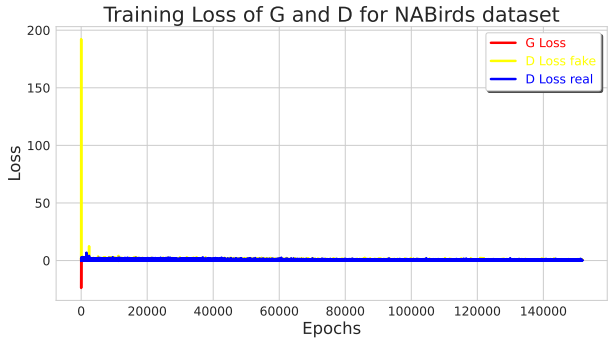
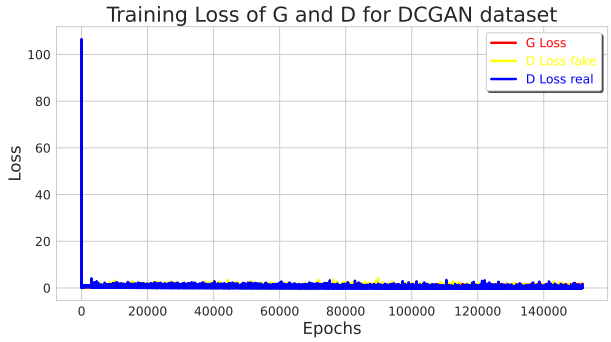


Figure 4: Translated samples to another style for Foods dataset.



(a) The graph of original loss generated for training the DeepI2I GAN using the NABirds dataset.



(b) The graph of loss generated for training the DeepI2I GAN using the NABirds dataset using SoftPlus loss.

Figure 5: The losses generated by training NABirds dataset.

for this dataset and the GAN was trained from scratch, a learning rate of 1e-04 was used for the Generator and a learning rate of 4e-04 was used for the discriminator. The results were below satisfactory, given that there is a lot of issues while training GANs.

4 Data set Details

For this experiment, we have used three datasets, and have performed comparative analysis on all of them:

- **Animal faces dataset** - This dataset contains 117,574 images and 149 classes in total. This dataset was used in the FUNIT project (`https://github.com/NVlabs/FUNIT`) and the animal faces were cropped out from the ILSVRC [8] dataset.
- **Foods dataset** - This dataset consists of 31,395 images and 256 classes in total. This dataset can be retrieved from here `http://foodcam.mobi/dataset256.html`.
- **NABirds (North America Birds) dataset** - This dataset has 48,527 images and 555 classes in total. The dataset can be retrieved from `https://dl.allaboutbirds.org/nabirds`.

All these datasets are image datasets containing 3-channel RGB images and a corresponding label associated with them as class values. All these images are of different shapes and sizes, but we have resized them to 128x128 pixels and divided by 255.0 to normalize the intensity before passing to the model. For each of them, we have split 90% of the dataset as a training set and 10% as a test set. The datasets were processed to .HDF5 files for faster I/O and batch-sized pre-processing. The pipeline of the code is borrowed from BigGAN architecture. We have used Python3 and Pytorch1.12.1 framework for this study.

5 Experiments

When we trained the model for NABirds dataset we obtained a RC of 3.24, FC of 5.84 and a mKIDx100 of 30.5. This was trained using the deepI2I model from scratch. During the training, a batch size of 4 was used for all the datasets, and the GAN was trained from scratch, a learning rate of 1e-04 was used for the Generator and a learning rate of 4e-04 was used for the discriminator.

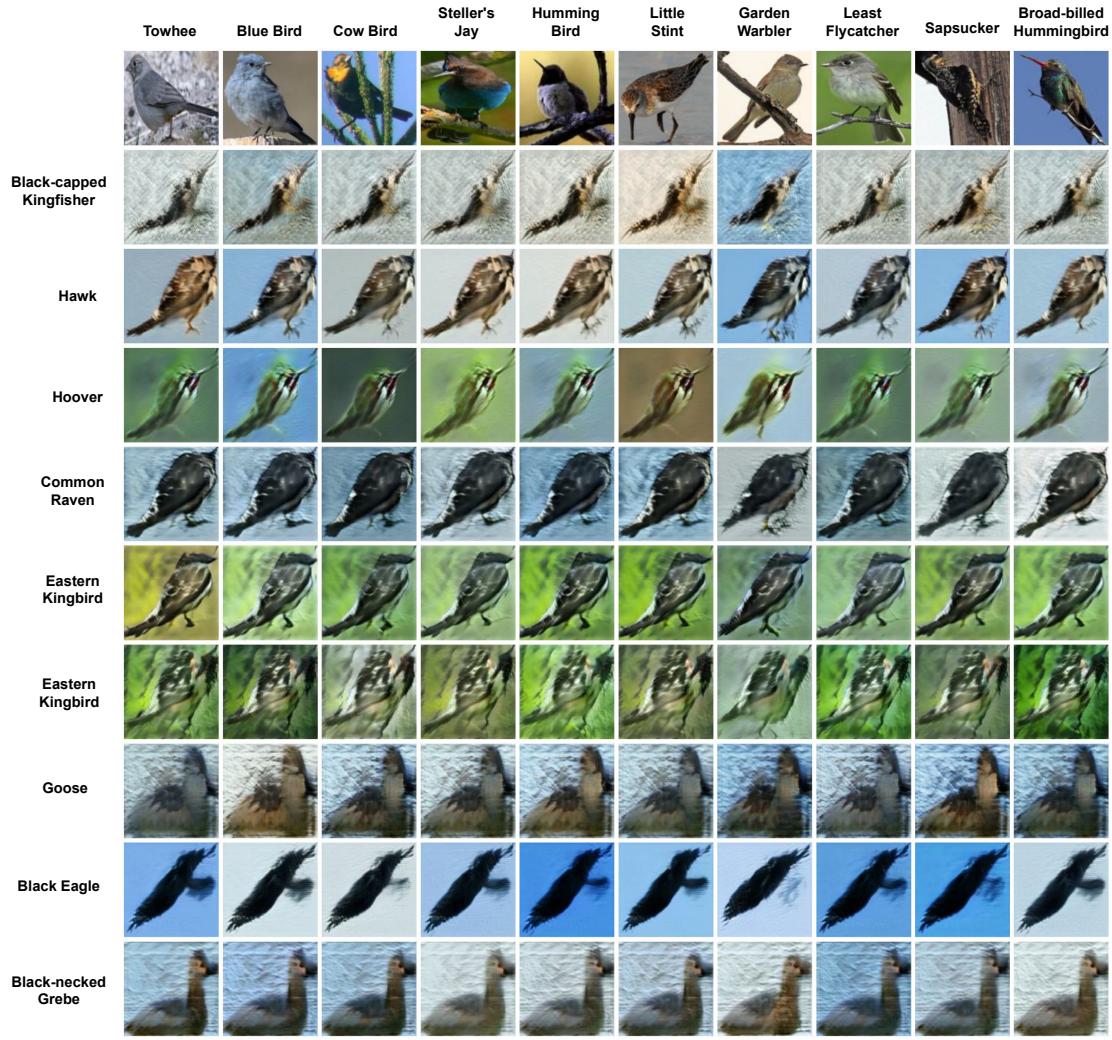


Figure 6: Translated samples to another style for NABirds dataset.

(a) Few samples generated from NABirds dataset.

(b) Few samples generated from NABirds dataset.

Figure 7: Random samples are taken from the Generator and projected to similar latent space (Z=120 here).

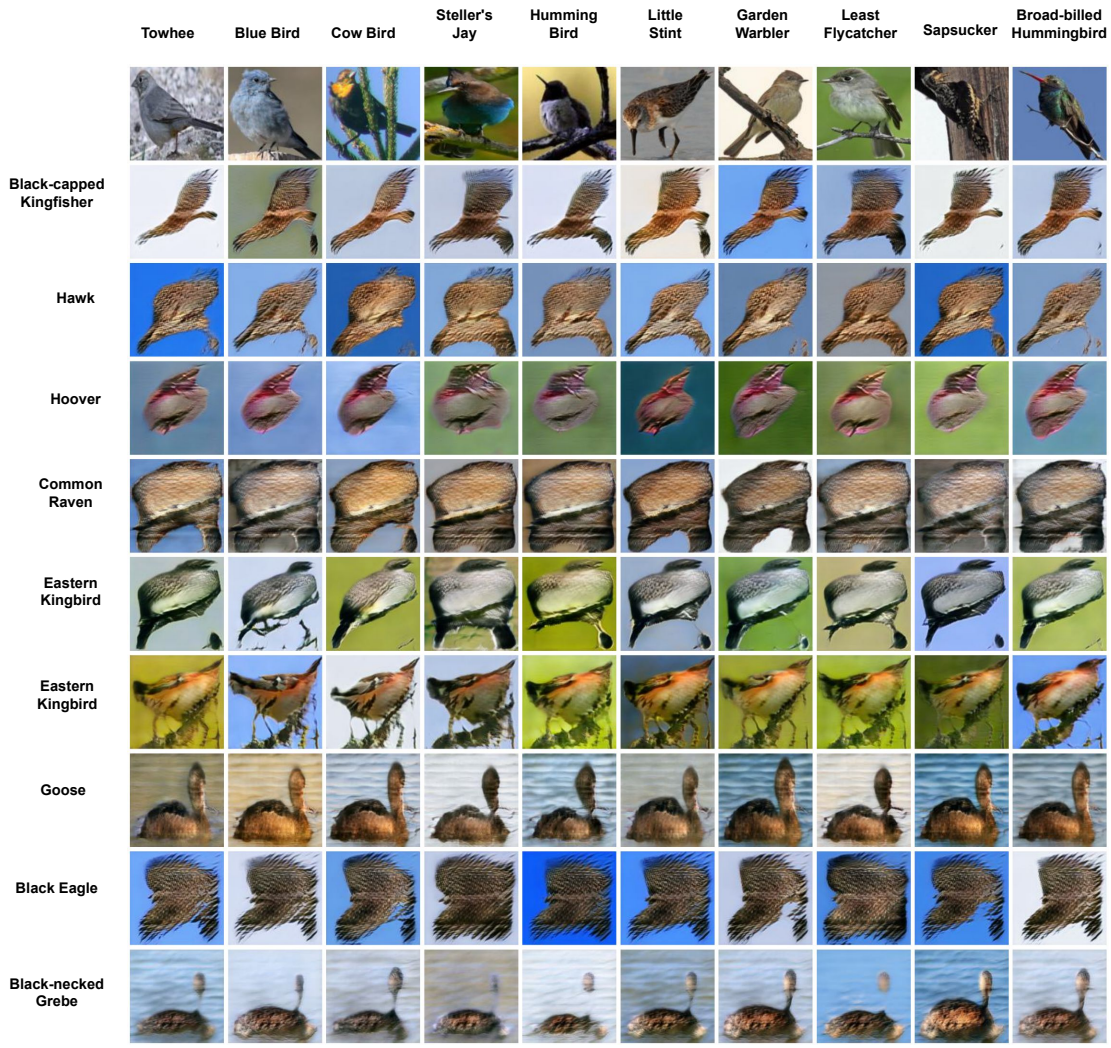
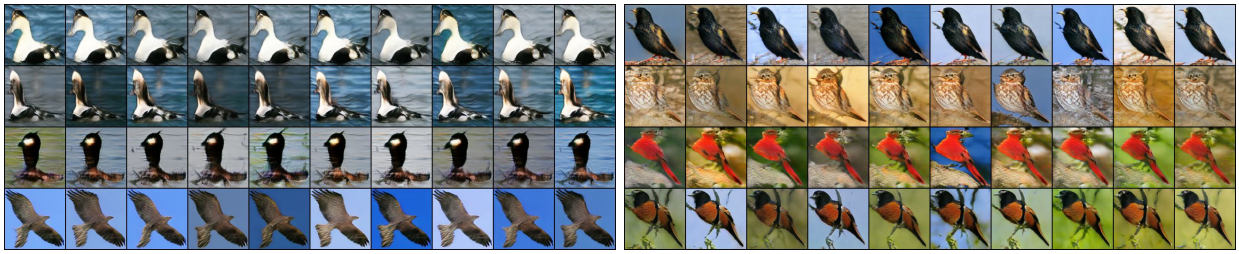


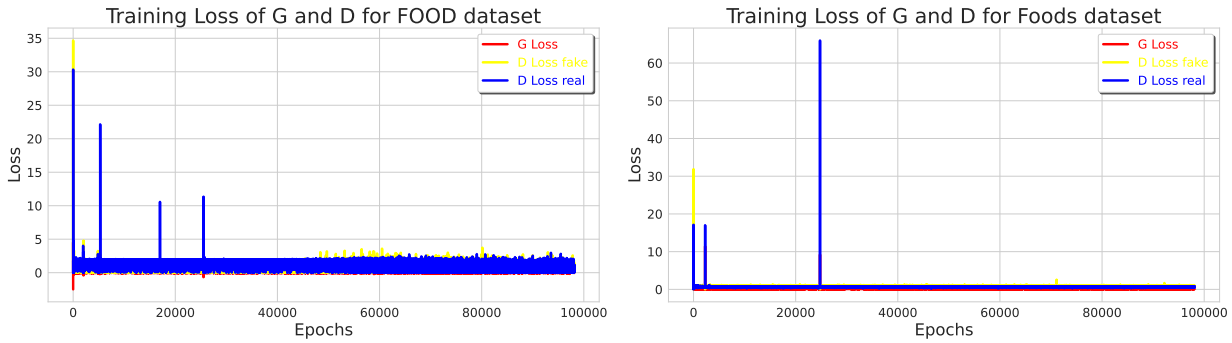
Figure 8: Translated samples to another style for NABirds dataset using DCGAN loss.



(a) Few samples generated from food dataset.

(b) Few samples generated from NABirds dataset.

Figure 9: Random samples are taken from the Generator and projected to similar latent space (Z=120 here), by using the SoftPlus loss function.



(a) The graph of original loss generated for training the DeepI2I GAN using the Foods dataset.

(b) The graph of loss generated for training the DeepI2I GAN using the Foods dataset using SoftPlus loss.

Figure 10: The losses generated by training Foods dataset.

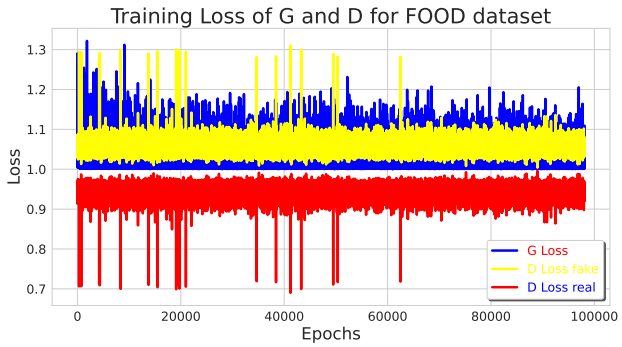


Figure 11: The losses generated by training Foods dataset using SSIM loss.

(a) Few samples generated from food dataset using SoftPlus loss, a result of mode collapse.

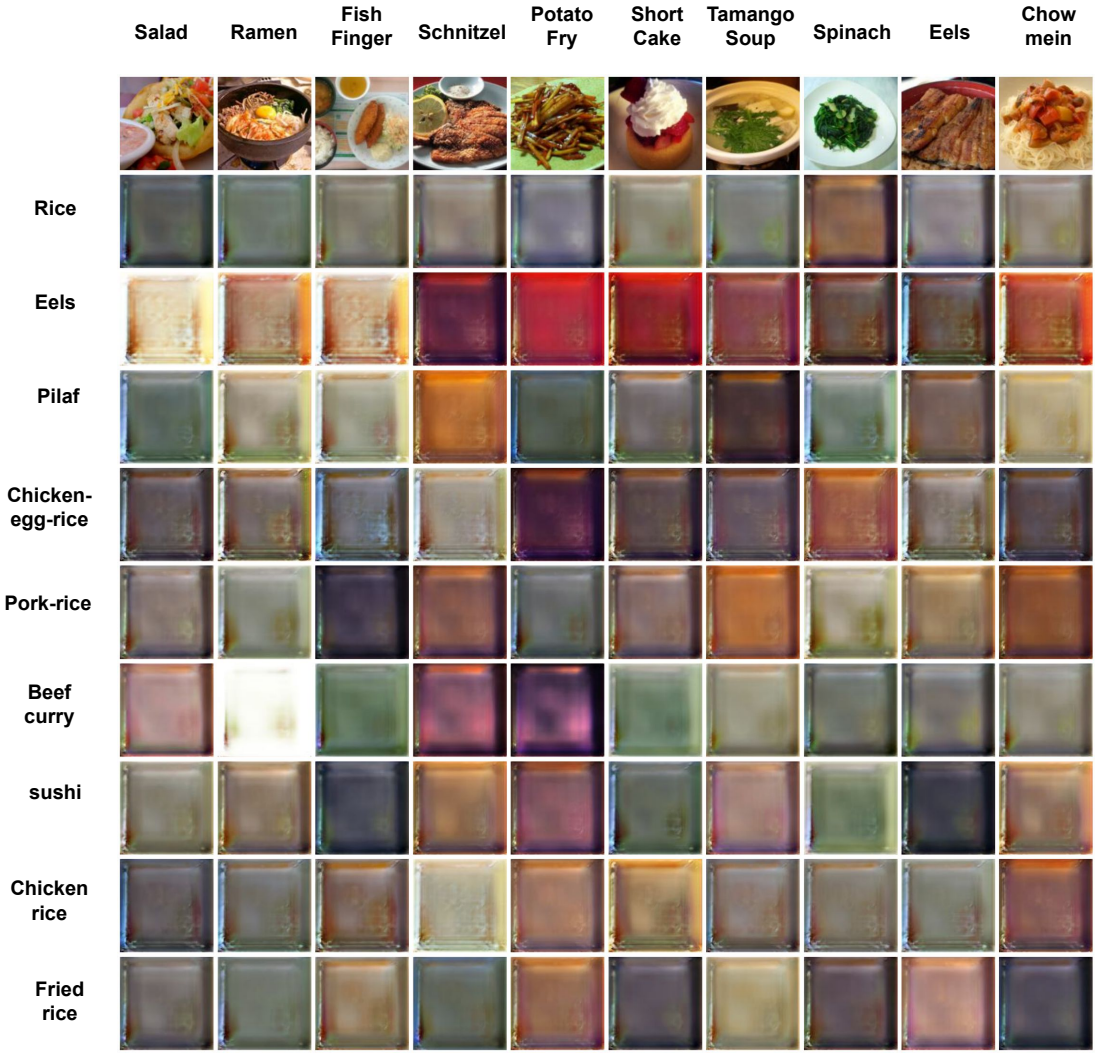
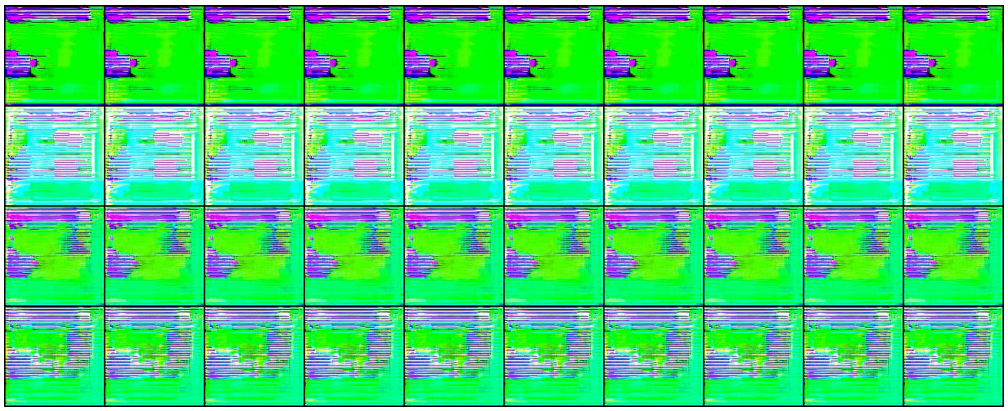


Figure 13: Translated samples to another style for Foods dataset, GAN unable to generate quality images.



(a) Few samples generated from food dataset using SSIM loss, which makes the GAN unable to generate the samples.

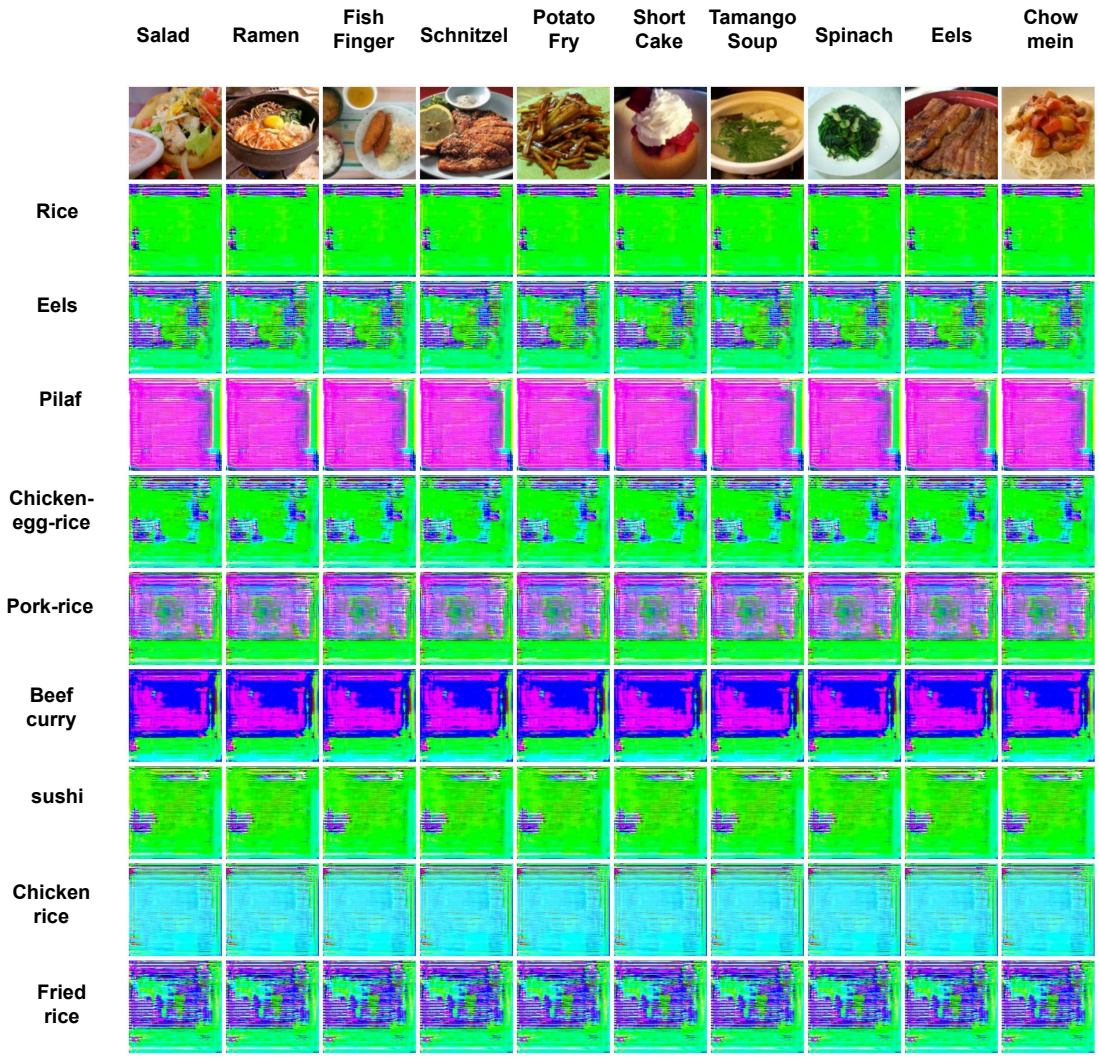
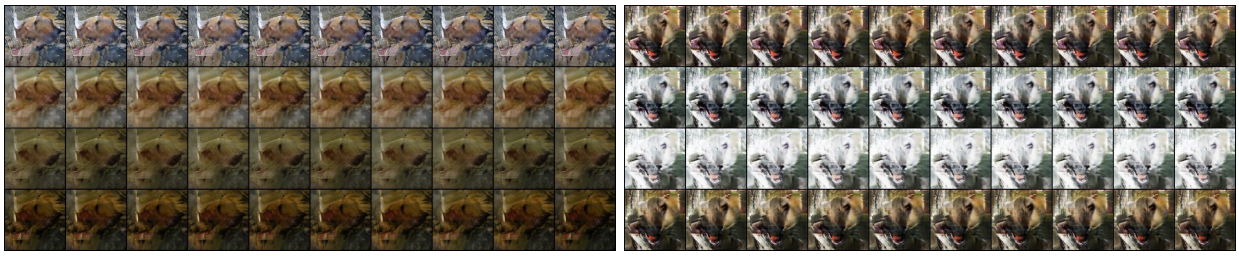


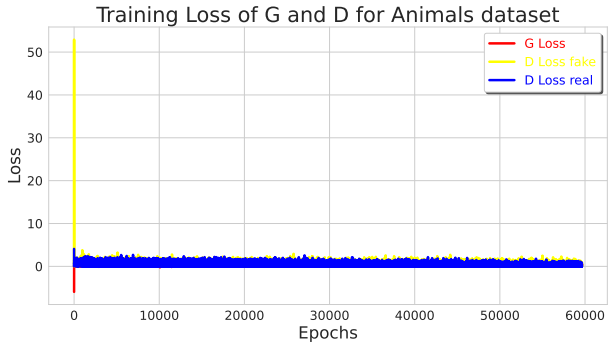
Figure 15: Translated samples to another style for Foods dataset, GAN unable to generate images.



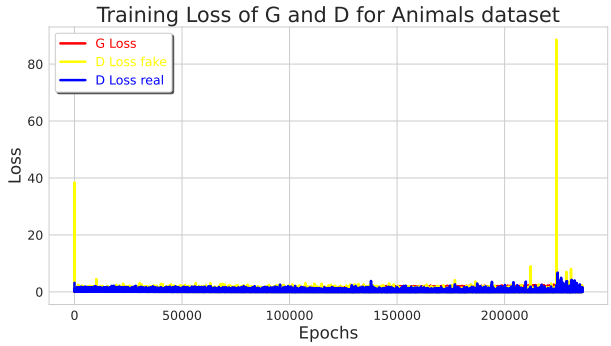
(a) Few samples generated from Animals dataset using normal loss.

(b) Few samples generated from Animals dataset using Softplus loss.

Figure 16: Random samples are taken from the Generator and projected to similar latent space (Z=120 here), by using the SoftPlus loss function.



(a) The graph of original loss generated for training the DeepI2I GAN using the Animals dataset.



(b) The graph of loss generated for training the DeepI2I GAN using the Animals dataset using SoftPlus loss.

Figure 17: The losses generated by training Animals dataset.

Most of the training and re-training was done after the midterm review. For instance, we got some really good samples for the foods dataset as shown in Figure 3. The training graph is shown in Figure 10a. The translated samples are shown in Figure 4. Same dataset was trained using SSIM loss and the resulting samples were shown in Figure 14a. The graph and translated samples are shown in Figure 11 and 15. When we train the dataset with the SSIM loss, we can see that the model is not able to produce any meaningful images, hence we didnot use SSIM loss. The dataset when trained with Softplus loss gives mode collapse and the samples, loss graph and translated samples are shown in Figure 12a, 10b, and 13.

For NABirds dataset we got relatively good samples as shown in Figure 7b. The graph and class transition are shown in Figure 5a and 6. The use of softplus loss gave a better result in terms of visual quality as shown in Figure 9. The graph and the class transitions are shown in Figure 5b and 8.

For the Animals dataset we got mode collapsed samples as shown in Figure 16a. The graph and class transition are shown in Figure 17a and 18. The use of softplus loss gave a similar mode collapse result as shown in Figure 16b. The graph and the class transitions are shown in Figure 17a and 19.

5.1 Hardware details

The hardware for this project was generously provided by my guide, Prof. Suyash P. Awate. The system consists of 8 x NVIDIA GeForce RTX 2080 Ti GPUs with Intel Xeon Gold 6130 @ 64x 2.101GHz processor, 5.4 TB space of solid-state drive, Ubuntu 18.04 LTS Operating system and a main memory of 128 GB (RAM).

6 Results

The final results are shown in Table 1. We can see we have achieved marginally better results than the previously trained NABirds dataset, and most of the re-training resulted in mode collapse problem in GANs. This is a serious issue for GANs and there are current studies which extensively deals with them. For our part we were sucessfully able to reproduce the results of the dataset for animals and foods as present in the paper as shown in Table **??**.

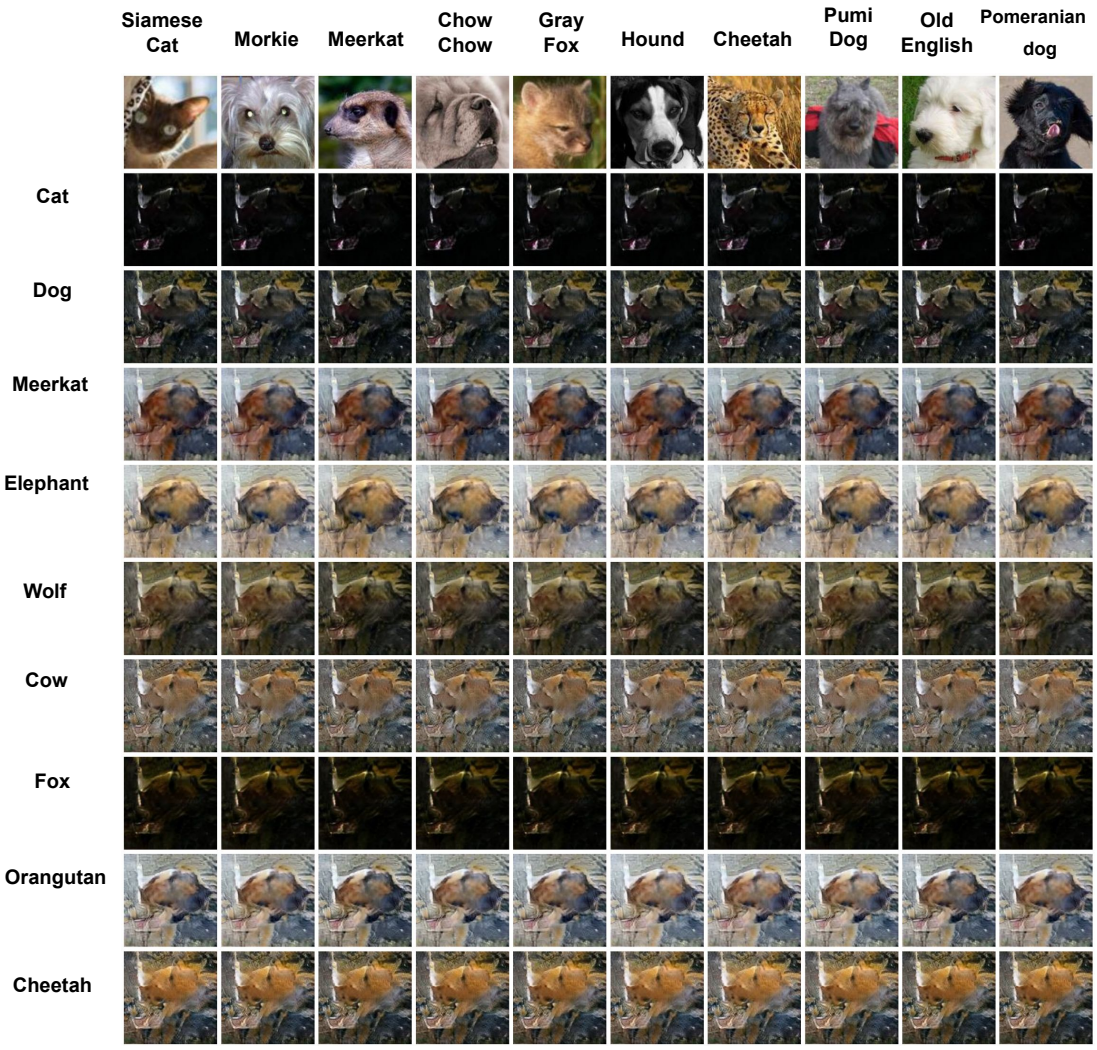


Figure 18: Translated samples to another style for Animals dataset using normal loss.

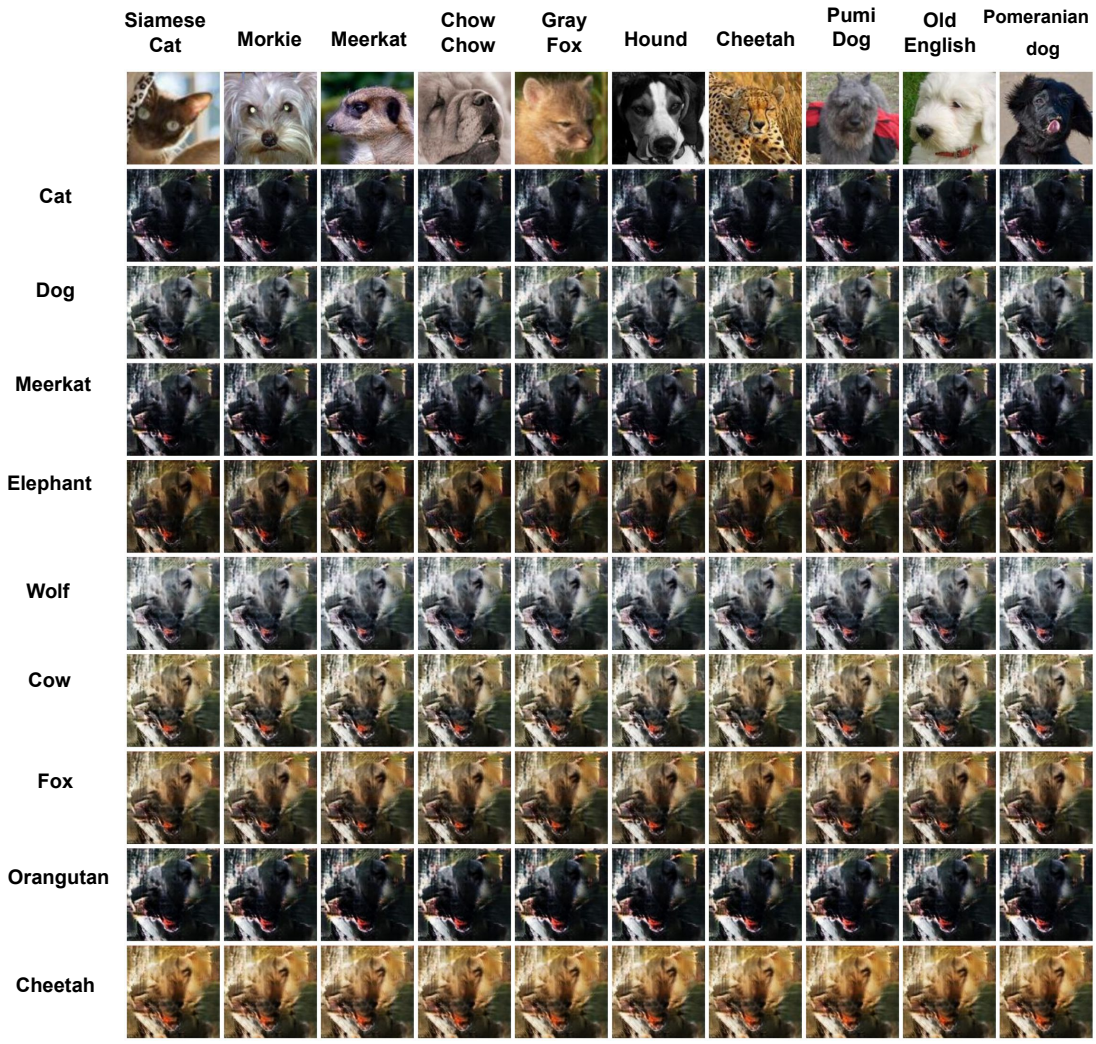


Figure 19: Translated samples to another style for Animals dataset using softplus loss.

	RC ↑	**FC ↑**	**mKIDx100 ↓**	**mFID ↓**
Animal (Ori. Loss)	49.2	52.4	5.78	80.7
Food (Ori. Loss)	5.83	4.67	26.5	278.2
Birds (Ori. Loss)	3.24	5.84	30.5	301.7
Birds (Our Loss - SoftPlus)	**3.57**	**5.93**	**30.71**	**301.9**

Table 1: Final results as reproduced by us in this work.

Method \ Datasets	Animal faces (*710/per class*)				Foods (*110/per class*)			
	RC↑	FC↑	mKID×100↓	mFID↓	RC↑	FC↑	mKID×100 ↓	mFID↓
StarGAN	33.4	38.2	15.6	157.7	10.7	12.1	20.9	210.7
SDIT	32.9	39.1	15.3	151.8	11.9	11.8	23.7	236.2
DMIT	36.7	42.1	14.8	146.7	8.30	10.4	19.5	201.4
DeepI2I (scratch)	49.2	52.4	5.78	80.7	5.83	4.67	26.5	278.2
DeepI2I	**49.5**	**55.4**	**4.93**	**68.4**	**30.2**	**19.3**	**6.38**	**130.8**

Table 1: Showing the comparison of different results for Animals and Foods datasets, replicated from the original papers.

7 Future Work

The SSIM and SI-SDR loss functions can be tried with VAE based Generative networks. The current loss functions can be fine tuned for better quality visual results by hyper-parameter tuning, but that needs a lot of tuning to get comparable results with different loss as reported in the paper. The quality of the images could be increased more, like 1080 x 1080 px, by using different up-sampling GAN architectures.

8 Conclusion

Training GANs can be very difficult. Even carefully tuned hyper-parameters can lead to mode collapse during prolonged-training. The general scheme in overcoming such kinds of problems is to save and evaluate models after succession of certain time periods. Building loss function for GANs is a difficult task, since in theory, it may seem to work, but in general it might not work without carefully fine-tuned hyper-parameters.

In this work, we have successfully implemented a DeepI2I architecture, by reproducing the original results presented by the authors, used the model to train on a different dataset and created a few loss functions. We have seen that these types of results and models can be used for different graphics industries, where they could replace the manual labor by using a model. These models not only provide fast solutions, but also accurate solutions, if trained carefully. The original codebase uses hinge loss type objective, we have modified it to use a DCGAN type objective, which gives visually better results. We also propose future directions for this work, which might use different architecture in general, like VAEs, to experiment on more types of loss functions.

References

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[2] Yunjey Choi, Min-Je Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8789–8797. Computer Vision Foundation / IEEE Computer Society, 2018.

[3] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014.

[4] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 179–196, Cham, 2018. Springer International Publishing.

[5] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5967–5976. IEEE Computer Society, 2017.

[6] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019.

[7] Hsin-Ying Lee, Hung-Yu Tseng, Qi Mao, Jia-Bin Huang, Yu-Ding Lu, Maneesh Singh, and Ming-Hsuan Yang. DRIT++: diverse image-to-image translation via disentangled representations. *Int. J. Comput. Vis.*, 128(10):2402–2417, 2020.

[8] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[9] Yaxing Wang, Abel Gonzalez-Garcia, David Berga, Luis Herranz, Fahad Shahbaz Khan, and Joost van de Weijer. Minegan: Effective knowledge transfer from gans to target domains with few images. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9329–9338. Computer Vision Foundation / IEEE, 2020.

[10] Yaxing Wang, Abel Gonzalez-Garcia, Joost van de Weijer, and Luis Herranz. SDIT: scalable and diverse cross-domain image translation. In Laurent Amsaleg, Benoit Huet, Martha A. Larson, Guillaume Gravier, Hayley Hung, Chong-Wah Ngo, and Wei Tsang Ooi, editors, *Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21-25, 2019*, pages 1267–1276. ACM, 2019.

[11] Xiaoming Yu, Yuanqi Chen, Shan Liu, Thomas H. Li, and Ge Li. Multi-mapping image-to-image translation via learning disentanglement. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 2990–2999, 2019.

[12] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on Computational Imaging*, 3(1):47–57, 2017.

[13] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2242–2251. IEEE Computer Society, 2017.