Relativistic Discriminator Based Unsupervised Pose Guided Human Image Generation Using GAN

Merga W/yohannes Enkossa

A Thesis Submitted to the Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master's in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

August 2021

Adama, Ethiopia

RD-UPGHIG: Relativistic Discriminator Based Unsupervised Pose Guided Human Image Generation Using GAN

Merga W/yohannes Enkossa

Advisor

Worku Jifara Sori (Ph.D)

A Thesis Submitted to the Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in partial fulfillment of the Requirement for the Degree of Master's in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

August 2021

Adama, Ethiopia

Declaration

I hereby declare that this Master Thesis entitled "**Relativistic discriminator based unsupervised pose guided human image generation using GAN** " is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation

Merga W/Yohannes Enkossa _____ _____

Name Signature Date

Recommendation

I, the advisor of this thesis, hereby certify that I have read the revised version of the thesis entitled "**Relativistic discriminator based unsupervised pose guided human image generation using GAN**" prepared under my guidance by Merga W/Yohannes Enkossa submitted in partial fulfillment of the requirements for the degree of Master's of Science in Computer Science and Engineering. Therefore, I recommend the submission of a revised version of the thesis to the department following the applicable procedures.

Dr. Worku Jifara (Ph.D)	_____	_____
Major Advisor	Signature	Date
_____	_____	_____
Co-Advisor	Signature	Date

Approval page

I, the advisors of the thesis entitled "**Relativistic discriminator based unsupervised pose guided human image generation using GAN**" and developed by Merga W/Yohannes Enkossa, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Dr. Worku Jifara (Ph.D)	_____	_____
Major Supervisor	Signature	Date
_____	_____	_____
Co-Supervisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis by Merga W/Yohannes Enkossa have read and evaluated the thesis entitled "**Relativistic discriminator based unsupervised pose guided human image generation using GAN** " and examined the candidate during the open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

_____	_____	_____
Chair Person	Signature	Date
_____	_____	_____
Internal Examiner	Signature	Date
_____	_____	_____
External Examiner	Signature	Date

Finally, approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

_____	_____	_____
Chair Person	Signature	Date
_____	_____	_____
Internal Examiner	Signature	Date
_____	_____	_____
External Examiner	Signature	Date

ACKNOWLEDGMENT

First of all, I would like to thank the Almighty God, for keeping me healthy and making me reach this milestone after many obstacles.

Then**,** I am very grateful to my advisor Dr. Worku Jifara (Ph.D.) who were helping me from the proposal to the completion of this research work, giving me the right directions and guidance, useful comments, valuable support, and supervision.

Next, I would like to thank Prof. Yunkoo Chung (Ph.D.) who were supporting me throughout my work in giving me comments, suggestions, and evaluations on my work. Also, I want to thank Mr. Anteneh Tilaye (Msc) for his supportive comments and suggestions by reading my thesis.

I would also like to express my deepest thanks to the computer vision and robotics SIG members and all my classmates for their supportiveness throughout my work. Finally, I would like to thank my families who were there in all my ups and down supporting and advising me, and my friends who have been supportive of my work and giving me advice throughout my thesis work.

TABLE OF CONTENTS

ACKNOWLEDGMENT i

LIST OF TABLES vii

LIST OF FIGURES viii

LIST OF SAMPLE CODES x

LIST OF ACRONYMS AND ABBREVIATIONS xi

LIST OF NOTATIONS AND SYMBOLS xii

ABSTRACT xiii

CHAPTER ONE 1

1. INTRODUCTION 1

1.1 Background of the Study 1

1.2. Motivation of the study 2

1.3. Statement of the Problem 3

1.4. Research question 3

1.5. Objectives 4

1.5.1 General Objectives 4

1.5.2 Specific Objectives 4

1.6. Scope and Limitations 4

1.6.1 Scope of the study 4

1.7.2 Limitation 4

1.7. Application of the Research 5

1.8. Organization of the Thesis 5

CHAPTER TWO 7

2. LITERATURE REVIEW AND RELATED WORKS 7

2.1. Introduction 7

2.2 Generative Adversarial Network 8
2.2.1 GAN Training 9
2.2.2 CycleGAN 10
2.3. Preprocessing and Feature Extraction Techniques 11
2.3.1. Preprocessing of the Data 11
2.3.2. Feature Extraction 11
2.4. Variational Auto Encoder 11
2.5. Image to Image Translation 12
2.6. Style transfer 14
2.7. Generating image of people in clothing 16
2.8. Related Works 17
2.9. Summary of Related Works 19
CHAPTER THREE 21
3. RESEARCH METHODOLOGY 21
3.1. Chapter Overview 21
3.2. Dataset Collection 21
3.2.1. Data Augmentation 22
3.3. Development Tools 24
3.3.1. Hardware Tools 24
3.3.2. Software tools 24
3.5. Baseline works 25
3.6. Feature extractor 25
3.7. Evaluation Methods 26
3.7.1. Human Evaluation Study 26
3.7.2. Inception Score 26

CHAPTER FOUR 27
4. PROPOSED ARCHITECTURE 27
4.1. Chapter Overview 27
4.2. Pre-processing Techniques 27
4.2.1. Normalization 27
4.2.2. Resize Image 27
4.2.3. Data Augmentation 27
4.3. Proposed Aproach Architecture 28
4.3.1. Feature Extractor 29
4.3.2. Generator architecture 32
4.4.3. Attention mechanism 33
4.4.3. Relativistic Discriminator 34
4.4. Model Learning Functions 36
4.4.1 Cycle compliment loss 37
4.4.2. Color cycle-consistency loss 38
4.5. Training Pseudocode 40
CHAPTER FIVE 41
5. IMPLEMENTATION OF THE PROPOSED SOLUTION 41
5.1. Chapter Overview 41
5.2. Working Environments 41
5.3. Environment Setup 41
5.4. Training Procedure 42
5.5. Implementation Techniques for Dataset Preprocessing 42
5.6. CycleGAN and VAE implementations 44
5.6.1. Color preserving loss 46

5.6.2. Cycle compliment loss 46
5.6.3. Relativistic average discriminator 47
5.7. Learning hyperparameters 47
5.8. Experiment Class 48
CHAPTER SIX 50
6. RESULT, EVALUATION, AND DISCUSSION 50
6.1. Chapter Overview 50
6.2. Results of the study 50
6.2.1. Dataset Preparation Result 50
6.2.2. Results of the Data Augmentations 51
6.3. Experimental Results 51
6.3.1. UPG-GAN 52
6.3.2. with cc + cccl 52
6.3.2. With LSGAN 53
6.3.3. RD-UPGHIG 54
6.4. Sample training log for RD-UPGHIG 58
6.5. Evaluation Metrics 59
6.5.1. Inception Score 59
6.5.2. Human evaluation 59
6.6. Discussion 61
6.6.1. Discussion on inception score results 61
6.6.2. Discussion for human evaluation 63
CHAPTER SEVEN 64
7. CONCLUSION AND FUTURE WORKS 64
7.1. Conclusion 64

7.2. Future Work 65
References 66
Appendixes 71
Appendix A: Sample of the datasets 71
Appendix B: Training log curves 72
Appendix C: Results of training at different epochs 75
Appendix D: Sample codes of the model 77
Appendix E: Human evaluation study form and results 79
Appendix E.1 Human evaluation google form snapshots 79
Appendix E.2 Human evaluation results 84

LIST OF TABLES

Table 2. 1: Summary of related works 19
Table 3. 1 : dataset before and augmentation 22
Table 3. 3: hardware tools 24
Table 3. 4: software tools 24
Table 5. 1: generator architecture of CycleGAN 45
Table 5. 2: contents of discriminator architecture 45
Table 5. 3: learning hyperparameters 48
Table 5. 4: Lists of experimental classes 49
Table 6. 1: Inception score results for shirt and t-shirt and suit and dress 59
Table 6. 2: Human evaluation summary on shirt and t-shirt 61
Table 6. 3:Human evaluation results of suit and dress 61

LIST OF FIGURES

Figure 1. 1: Application on virtual try-on 5
Figure 2. 1: GAN Architecture 9
Figure 2. 2: CycleGAN training 11
Figure 2. 3: VAE input image representation sample 12
Figure 2. 4: Results of an image to image translation from(Isola et al., 2017) 13
Figure 2. 5: Dataset specifications of pix2pix (paired) and CycleGAN (unpaired) 14
Figure 2. 6: Style transfer 15
Figure 2. 7: The control methods of style transfer 16
Figure 3. 1: Overall data acquisition process 21
Figure 3. 2: Sample data augmentation with the different techniques used 23
Figure 3. 3: Sample dataset images 23
Figure 4. 1: residual block 30
Figure 4. 2: Proposed architecture 31
Figure 4. 3: The Generator architecture 33
Figure 4. 4: SAGAN attention mechanism 34
Figure 4. 5 The generator's goal with, left: Relativistic GAN and, right: standard GAN 35
Figure 4. 6: Discriminator architecture 36
Figure 4. 7: left: cycle consistency loss and right: complementary cycle consistency loss 37
Figure 4. 8: Sample architecture showing the losses and generators 39
Figure 5. 1: Sample augmented dataset 43
Figure 6. 1: Dataset augmentation results 50
Figure 6. 2: Sample shirt and t-shirt augmented samples 51
Figure 6. 3: Results of UPG_GAN 52
Figure 6. 4: Results with the addition of cc and cccl 53
Figure 6. 5: Results of suit and dress with the addition of LSGAN 54
Figure 6. 6: Results of RdUPGHIG for suit 55
Figure 6. 7: Random image generated with RdUPGHIG 56
Figure 6. 8: The generation results of the different experiments (comparison and contrast) 57
Figure 6. 9: Training loss of generator for suit and dress of RdUPGHIG 58
Figure 6. 10: Training loss of discriminator for suit and dress of RdUPGHIG 58

Figure 6. 11: Google form question of human evaluation 60
Figure 6. 12: Graph of inception score for shirt and t-shirt 62
Figure 6. 13: Graph of inception score for suit and dress 62

LIST OF SAMPLE CODES

Algorithm 1: data augmentation algorithm 28
Algorithm 2: overall training algorithm during training 40
Code_snippet 1: code for augmenting the datasets 43
Code_snippet 2: code for defining generator and discriminator 44
Code_snippet 3: code for color consistency loss 46
Code_snippet 4: code for complementary loss 47
Code_snippet 5:code for defining relativistic discriminator 47

LIST OF ACRONYMS AND ABBREVIATIONS

2D	Two Dimensional
3D	Three Dimensional
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CycleGAN	Cycle consistent Generative Adversarial Network
DCGAN	Deep Convolutional Generative Adversarial Network
Def-GAN	Deformable Generative Adversarial Network
E-Commerce	Electronic Commerce
FID	Fréchet Inception Distance
GAN	Generative Adversarial Networks
GPU	Graphical Processing Unit
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IS	Inception Score
Kernel-MMD	Kernel-Maximum Mean Discrepancy
KNN-score	K-Nearest Neighbor score
LPIPS	Perceptual Image Patch Similarity
RAM	Random Access Memory
ResNet	Residual Network
ReLU	Rectified Linear Unit
SGAN	Standard Generative Adversarial Network
SSIM	Structural Similarity Index Metrix
VAE	Variational Autoencoders
UPG-GAN	Unpaired Pose Guided – Generative adversarial Network

LIST OF NOTATIONS AND SYMBOLS

A	input pose image
B	input Real image
D_x	Discriminator for domain A
D_y	Discriminator for domain B
E	Expectation
GAB	Generator, for translating from domain A to B
GBA	Generator, for translating from domain B to A
I2I	Image to image
CC	Complementary cycle loss
CCCL	Color consistency cycle loss
RdUPGHIG	Relativistic discriminator based unsupervised pose guided human image generation
$\sim$	shows synthesized image
$\wedge$	shows reconstructed image
λ_1 , λ_2	loss weights
Z	latent space

ABSTRACT

Generative modeling is nowadays becoming popular as Generative Adversarial Networks and Variational Autoencoders are producing great results in image generation, one of which is to generate images of the human body from poses (3D sketches). However, most of the existing works are requiring to use paired set of datasets which are in some cases very hard to obtain and complex. To overcome this problem, one approach is to use an unsupervised mechanism for unpaired data sets. this research work aims at introducing generative modeling that generates a human image that gives control on the type of clothing and poses provided by the user in an end-to-end unsupervised manner using CycleGAN and Variational Autoencoder. Further, the previous works suffer in capturing the details of clothing colors and the overall generated image suffers from artifacts and getting the desired pose. So, to alleviate these problems this research work introduces color cycle consistency loss, complementary loss, and relativistic discriminator to the CycleGAN model to generate visually more realistic images. Moreover, the inception score of the proposed solution along with different experiments have been measured and the proposed solution has shown improvements in generating the desired image with better quality and pose information with the score of 3.55 and 3.01 respectivelly on the upper and full body . Further, human evaluation was held to evaluate the quality of the images being generated and it has shown the proposed solution generates better quality images with average of 75% and 66.64% respectively for the upper and full body. Finally, from the evaluation results, the added relativistic discriminator, color cycle consistency loss, and complementary loss have improved the quality of the image being produced.

Keywords: pose, CycleGAN, VAE, Generative modeling, relativistic discriminator,unsupervised image generation, GAN

CHAPTER ONE

1. INTRODUCTION

1.1 Background of the Study

Recently in the computer vision and deep learning area, there has been great interest in generative models for synthesizing images like, given poses of different styles as an input and generating a human image with clothes. That means being able to generate a photo-realistic image given that a pose or sketch of the person haves many applications in different areas like fashion and the E-Commerce industry (create photos of imaginary fashion models, with no need to hire a model), image editing, synthesizing training data for discriminative approaches in person detection, identification or pose estimation and the movie industry to name a few of them.

Many works on pose-guided human image synthesis are now being done with the advancement of deep learning and generative models (Xu Chen et al., 2019)(Ma et al., 2017)(Isola et al., 2017)(Lassner et al., 2017). Mostly the existing researches were based on paired or supervised which is limited because of its requirement for paired image dataset (Ma et al., 2017)(Lassner et al., 2017) which is hard to get in many cases for training purpose of the models and that's where the unsupervised or unpaired data come to solve the limitation and enable generating a flexible image. The recent generative models like variational autoencoders (VAE)(Esser et al., 2018), and generative adversarial networks (GANs) (Goodfellow et al., 2020) have significantly helped in producing the desired output.

Mostly the GAN methods are well known nowadays because of their capability to generate very realistic images through their adversarial training. For example, (Radford et al., 2016)(Xi Chen et al., 2016) GANs are used to generate images of faces and natural sceneries, with various additional approaches to train the process and increase the quality being offered.

Some of the current approaches try solving the problem of synthesizing images of people in clothing as a two-stage paired image-to-image translation task. In (Lassner et al., 2017) requires a pair of input images, which are the 3D body poses and their corresponding parsing images. Others also tried to achieve the transformation of the image from pose using two steps: (1) learning the important visual features of the person and (2) imposing the required pose on the person. But, most

of the state of art mechanisms that have been mentioned above give control on how to transform the given image into the target pose, they don't try to give control on the color, variety generation, or conditional sampling. Also, the quality of images that are generated from the real image is not that good, there are artifacts on their outputs.

Here, the key idea that is going to be done in this research will be generating of human realistic image conditioned on the target image and the provided pose image, giving the user the ability to control the pose, types of clothes, and the specific instance of the clothing with more accurate content with its corresponding target image. That means providing a pose and a real image the model should generate the respective real image in its respective target pose given with good color consistency and sufficient variety.

So, this proposed "relativistic discriminator based unsupervised pose guided human image generation" has tried to contribute further improvement on the dataset preprocessing and network architecture to give better results without requiring the paired datasets for training. So, for this work, the only required data are of two types that are the target real image and the target pose for a generation.

Further, this work has contributed technical level improvements with the addition of different mechanisms and constraints for improving the qualities and performance of the work.

1.2. Motivation of the study

Nowadays General Adversarial Network is producing a very realistic image that could be used in many areas for example in places when more datasets are required, in fashion and E-Commerce for virtual try-on, or could be used in places of the human models for advertisements, but getting much of this is hard. So, looking at this and many other applications of this GAN, motivated me to look forward to generating a human image that is based on the target pose given by the user. But in this GAN image generation getting paired data for training is expensive and difficult. To overcome these many mechanisms are proposed with GAN, one of which is using the unsupervised mechanism which works on the unpaired data's, this way we do not have to provide a paired data (during training) and this will give a great solution for those problems limited with data.

The other thing is that human beings can think of a person in particular poses, wearing specific types of clothes, so what if a machine learns to do those particular things with the GAN which is

nowadays generating very realistic images with high qualities. So that a machine can also try to generate different images according to a given pose

1.3. Statement of the Problem

Generating a human image with clothes has been mostly done with supervised (paired data) and in some cases getting the desired paired data is hard. Even in the unsupervised mechanisms, there are many challenges to capture the desired color, content, and styles of the clothes being generated. This problem could be because of the nature of the data being unpaired (the lack of the corresponding target image pair) and the absence of different optimization techniques for those specific problems (like color, style). So, most of the previously done researches lack in providing the contents of the given target image when synthesizing them and they don't give a clear content of the target image information on the output (Xu Chen et al., 2019). But in the fashion and E-commerce perspective, the quality of the image produced has a significant advantage in the eyes of the users so it needs to preserve the contents of the target image given, without color, style, and content loss.

In (Xu Chen et al., 2019) have attempted to utilize the CycleGAN with cycle consistency loss to overcome the problem of requiring a paired training data and other optimization procedures, but these procedures were not sufficient to provide the specified results. So, to overcome these challenging problems, Relativistic discriminator-based Unsupervised pose Guided Human image generation is proposed in this paper.

1.4. Research question

The following research questions were raised to be answered in this research:

RQ1: What could be the effect of adding content preserving constraints to the unpaired image generation?
RQ2: What is the effect of Relativistic discriminator in the unsupervised human image generation?
RQ3: What could be used to measure the performance of the work?

1.5. Objectives

1.5.1 General Objectives

The general objective of this study is to develop a relativistic discriminator-based human image generation from a given pose (3D sketch) using GAN in an unsupervised manner.

1.5.2 Specific Objectives

To address the general objective of the study, the following specific objectives were listed.

- To identify the constraints that can preserve the content loss (like color and shape) and details of the architecture.
- To improve the discriminator network to make it help generator training stable and generate quality images.
- To develop the architecture with the addition of improving loss functions.
- To train the model on the available dataset.
- To evaluate the performance of the model on the tested images using different metrics.

1.6. Scope and Limitations

1.6.1 Scope of the study

The main target of this research work is to generate a realistic-looking image of a person given that its pose. While generating its conditioned on the pose, target image, and optionally the class label (for generating a variety of images with different colors).

So, this study gives the following specific controls while generating an image:

- Control over the posture
- The type of cloth worn
- Conditional sampling that is generating a variety of images.

1.7.2 Limitation

Some of the limitations faced in the doings of these research work are listed as follows:

- It doesn't consider the temporal domain that means it will work in generating a single image.
- It doesn't give detail about the background of the generated image.

1.7. Application of the Research

The proposed research will have many applications in the following areas.

- In the fashion and E-commerce industry: it can be used as a human model in places of the real human model and also can be used in virtual try-on and others.

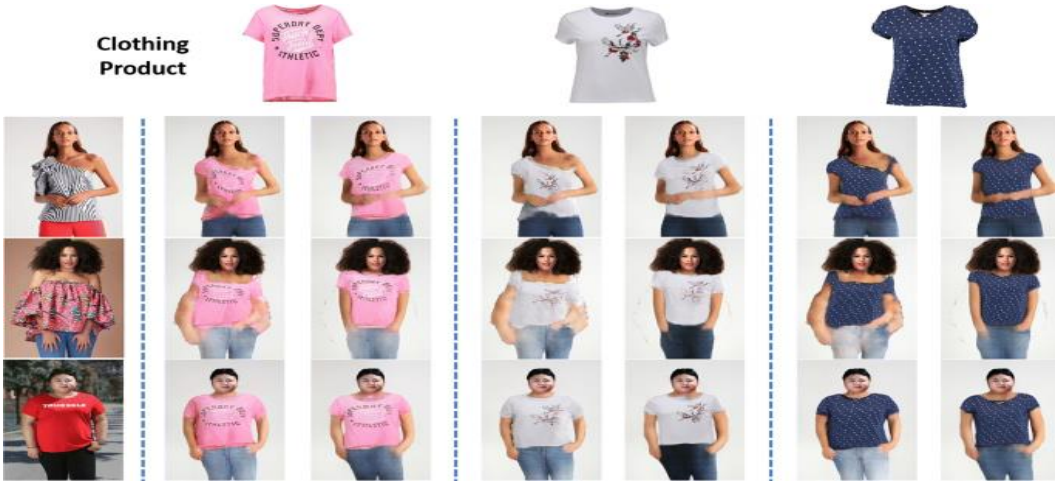


Figure 1. 1: Application on virtual try-on

- Synthesizing of training data in person detection and re-identification: usually in the re-identification problem the datasets are very limited hence it requires images from different angles and poses. With this method therefore we can generate training datasets for that purpose.
- Image editing
- Movie making industry

1.8. Organization of the Thesis

This research work is categorized into seven chapters and they are roughly described as follows.

Chapter one: This chapter introduces the overview of the research work and justifications how and what is going to be done.

Chapter two: This chapter discusses the literature and related works on image-to-image translation, generating people in clothing, and GAN's.

Chapter three: This chapter tries to give a clear description of how the research methodology is done, including the data collection techniques to the model evaluating metrics.

Chapter four: This chapter tries to give detailed information on how the proposed solution tries to solve the problem with different block diagrams, pseudo-codes, and flow charts of the implementation with corresponding mathematical descriptions.

Chapter five: This chapter explains the implementation of the proposed solution. Also, it provides the environment setup for the research and the code snippet of the proposed solutions.

Chapter six: This chapter discusses the results obtained from the proposed solution and gives a comparative description of the results with figures, graphs, and tables.

Chapter seven: This chapter gives the overall summary of the research. Gives the conclusion and the possible future works of the work.

CHAPTER TWO

2. LITERATURE REVIEW AND RELATED WORKS

2.1. Introduction

This chapter will explain in detail the generation of human images based on a given pose, different methods used to get the desired results in machine learning, results, limitations, and related works. The work of generating a human image is mostly done through learning and improving from their experience which is called machine learning.

Machine learning is an algorithm that builds a prediction model by learning from data and then when new data is given, it will predict based on what it has learned. Nowadays it is being widely used in different researches that provide services that we use in our daily activities like in Netflix, YouTube, Google, Facebook, and Twitter. This learning, now comes in three forms that are supervised, unsupervised, and reinforcement learning. Supervised learning uses labeled data to train the model for predicting future data. The labeled data are used for telling the machine what to look for while training. The supervised learning also tries to compare the output and the corresponding real (correct) one and tries to find the error which will help in updating the model for later on. Unsupervised learning does not have a labeled data for learning during training and it is mostly used in clustering. Here, when providing those data to the machine, it tries to identify the patterns of their performance and clusters them. Reinforcement learning is reward-based learning and works on a principle of feedback, that is when a machine is given input and generates the output according to the algorithm correctly it will be taken as correct else if it is wrong it will give feedback for the model to predict again. So, to sum it up, reinforcement learning is learned through trial and error.

Deep learning is part of machine learning inspired by how a human brain filters information and works. Most of its methods use neural networks and it is usually called a deep neural network. It has some fundamental architectures like a convolutional neural network, recurrent neural network. LeNet-5 is the first CNN introduced in 1989 by LeCun (Lecun et al., 1998) for recognition of handwritten digits and it was a breakthrough for many researchers for the efficient CNN models that we are using now.

Now in deep learning Generative model approach is being widely used. Generative Adversarial Network was first introduced by Ian Goodfellow in his 2014 paper(Goodfellow et al., 2020). The GAN model architecture is built of two sub-models, the generator model which will learn to generate a fake sample, and, in its adversary, The discriminator which will try to learn to distinguish the synthesized image as real or synthesized. Here, both the networks act as two neural networks competing to win, and this is explained in detail in the following section.

2.2 Generative Adversarial Network

As described above, it is a system or architecture where two neural networks are competing with one another to generate a variation in the data being generated. So, as its name describes (generative) means creating fake data, (adversarial) comes from the concept where the discriminator and the generator are competing for winning which means the generator would try to fool while the discriminator would try not to get fooled. They both work simultaneously to learn and train the complex model. The discriminator D(x) in GAN is a classifier that is used to discern between true and generated data by the generator and it can be any network architecture that is compatible with the classifier (model). The inputs of the discriminator are the real input and the generated image and its output is a probability from [0 – 1].

The generator G(z) learns to create a fake image that could fool the Discriminator to classify the output as real. The input to the generator is usually a random noise(z) and it tries to produce a meaningful output from it.

The general math's behind the GAN can be described as,

$$\min_G \max_D V(G, D = E_x \sim P_{data}[\log(D(x))] + E_Z \sim p_z[\log(1 - D(G(z)))] \quad (2.1)$$

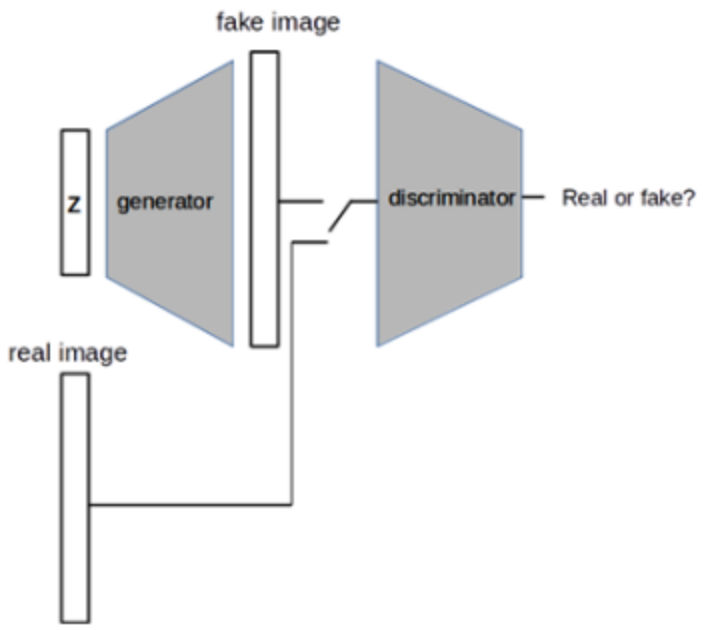


Figure 2. 1: GAN Architecture

(Source: cited from (Cheong, n.d.))

Even though the generator and discriminators are derived from a single formula in (eq 2.1)

2.2.1 GAN Training

The GAN training takes place in two phases that are first freezing the generator and training the discriminator which means the generator training will be held off and there will be only a forward pass, no backward propagation. Second, Freezing the discriminator and training the generator to produce a more realistic outcome that would fool the discriminator.

Simple steps for training a GAN model can be:

- Defining the problem and collecting the data
- Defining the architecture of the GAN to be used
- Training the Discriminator on real data so it will generate it as real for several times
- Generating the fake samples from the Generator
- Training the Discriminator on the fake samples
- Finally, train the Generator with the outputs of the discriminator for fooling the Discriminator.

2.2.2 CycleGAN

Usually in the image-to-image translation that means generating a synthetic version of a given image requires a large dataset that is mostly paired. But most of the time getting these paired data is difficult and expensive to prepare and that is where the CycleGAN comes as a solution.

CycleGAN is a mechanism for automatic translations without requiring a paired samples(Liu et al., 2017). The model of CycleGAN is trained in an Unsupervised mechanism using a set of images from the target and source image which do not have to be corresponding to each other. So, in this framework the two generators G: X →Y and F: Y→X try to generate a realistic-looking image that can fool the discriminators D(x) and D(y) into classifying the image synthesized by the generators as real. The CycleGAN tries to get to work first by using the first generator G to convert from one domain to another and then from the synthesized image to the original using the second generator F. Then it tries to calculate the mean squared error loss between the reconstructed image and the real image to check their similarities and further improve.

Cycle consistency loss: it is a base for CycleGAN that introduced the mechanism to translate an image from one domain to another without requiring a paired data during training. So, its idea is that when translating from one domain to the other domain (X2Y) and then back translating from that domain to the original (Y2X) we need to come to where we have started (the original). If not it needs to minimize the mean squared distance between the two images to better perform.

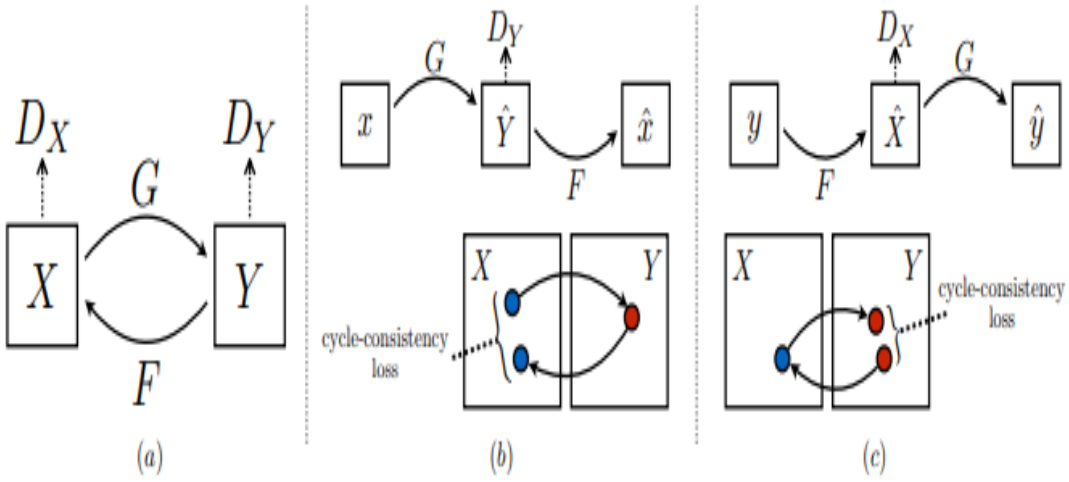


Figure 2. 2: CycleGAN training

(Source: cited from (Liu et al., 2017))

2.3. Preprocessing and Feature Extraction Techniques

2.3.1. Preprocessing of the Data

Data preprocessing is a crucial step in machine learning because the quality of information and the valuable data that can be determined from it will particularly affect the learning ability of our model; for that reason, before providing data to our model, we must preprocess it. Therefore, it is a raw data preparation technique for the construction and training of any machine learning model. And this section is briefly explained in section 4.2.

2.3.2. Feature Extraction

Feature extraction is a commonly used technique in machine learning and it is a dimensionality reduction process through which the contents of the original data are reduced into a new set of features that can describe the original data. So, the reduced set of features summarizes the content of the original features. Using this feature extraction usually helps in improved visualizability of the data, reduced overfitting problems, and model's improvement by accuracy. For this research, the VAEs encoder part is used to get the desired information from the images to further help the generation of the image with the detailed image information. And from different feature extraction techniques, those that correspond to this work were discussed in section 4.3.1.

2.4. Variational Auto Encoder

In deep learning nowadays variational autoencoder (VAE) and generative adversarial networks (GAN) are mobilizing the generation of images. VAE was the extension of an Autoencoder, and autoencoder is first introduced in (Rumelhart et al., 2013) for encoding input data through a neural network in an unsupervised manner to compress it to a smaller representation and then decode it. Therefore, VAEs are trained to reconstruct the input that it has taken. But autoencoders had many problems like regularization of the latent space which may result in overfitting. So, the variational autoencoder is the major improvement of the autoencoder (Kingma & Welling, 2014), It is a generative model, which is designed to regularize its training to overcome overfitting problems. The latent space in VAE is expressed as a distribution of learned mean and standard deviation to

give a good generation of the output. So, it will build an encoder to express the probability distribution of each latent space, instead of building an encoder that generates a unique value to describe each latent space. Figure 2.4 tries to describe the input image in a probabilistic term.

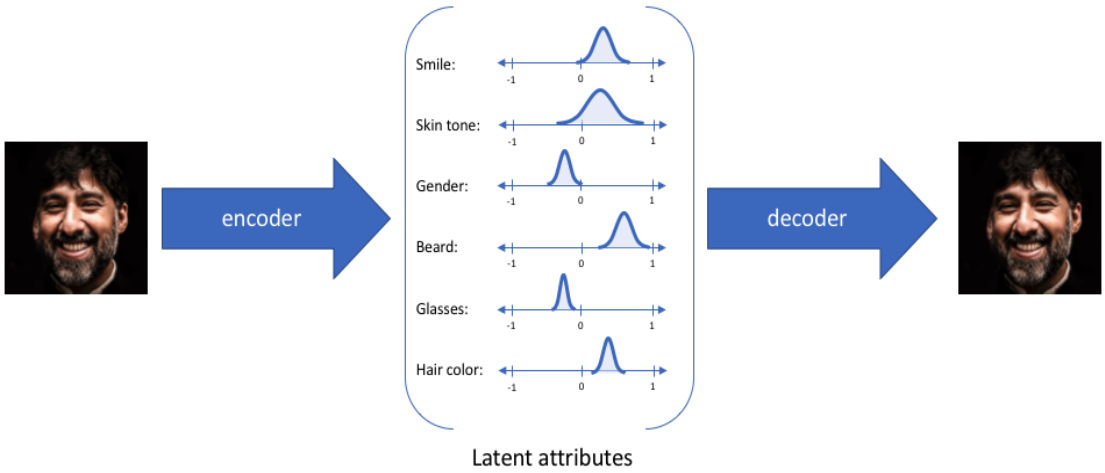


Figure 2. 3: VAE input image representation sample

(Source: cited from (Welling, 2017))

So, recently the VAE is being widely used with the GAN for the generation of images.

2.5. Image to Image Translation

I2I is a category of computer vision problems in which the aim is to figure out how to translate an input image to an output image. It may be used for a variety of tasks, including style transfer, object transformation, season translation, and photo enhancement. So, this problem was tackled in different ways using supervised learning (with paired sets of data) or unsupervised learning where the data for training is unpaired.

The majority of the currently available I2I translation is based on supervised learning that is they need paired data (Isola et al., 2017)(Liu et al., 2017) and for that reason, in many I2I translations scenarios, collecting paired training data is difficult and expensive. For example, (Isola et al., 2017) proposed a pix2pix framework, which can be used as a general solution for I2I conversion scenarios. For the training, pix2pix tries to combine the conditional GANs with the mostly used

loss function called pixel-wise L1 loss between the generated image and the ground truth image. The following shows the result from (Isola et al., 2017) paper for different datasets.

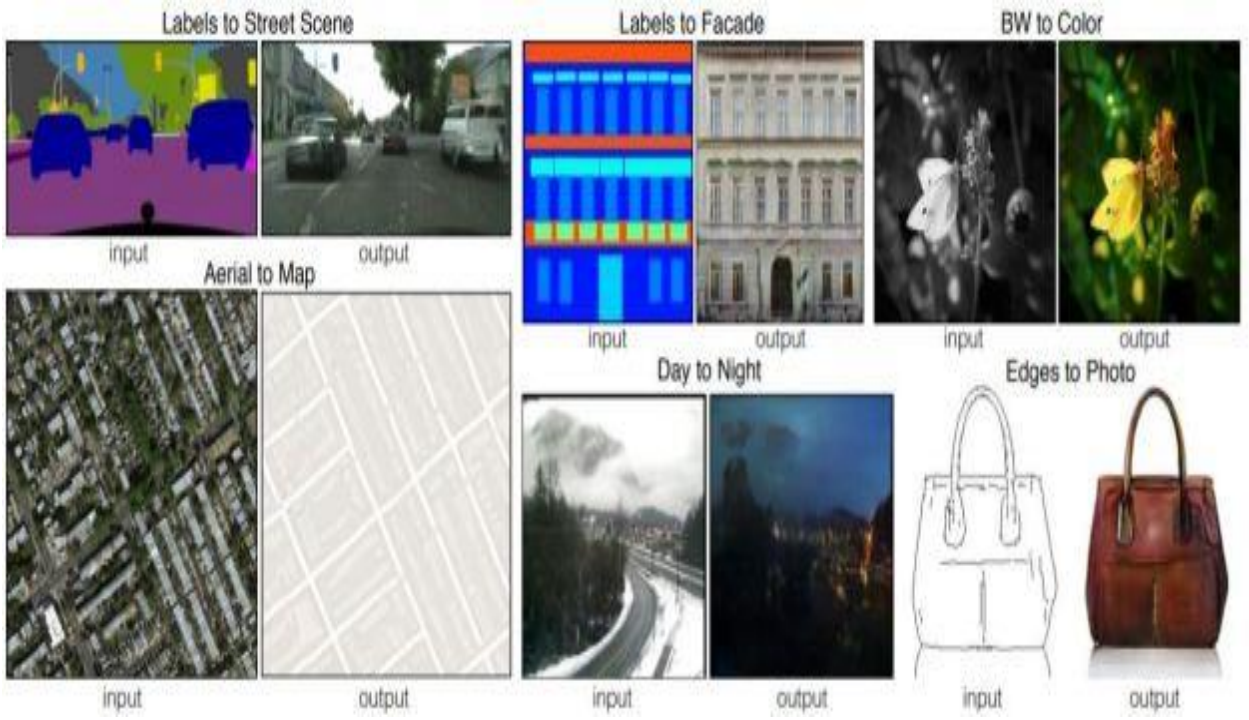


Figure 2. 4: Results of an image to image translation from(Isola et al., 2017)

(Source: cited from (Isola et al., 2017))

To overcome the problem of this paired data requirement, an unsupervised method called CycleGAN (Zhu et al., 2017) was introduced with cycle consistency loss between the original and reconstructed image. This technique tries to translate from one domain to the other domain without a one-to-one pairing between the two domains (that means without a paired data) as described in section 2.2.3. The problem with these methods is that they perform image-level translation, without considering the object instances and they tend to yield less realistic results when translating and also, hence the cycle consistency loss is calculated between the real and reconstructed image there is no relationship between the generated and real image which in turn results in less quality and artifacts on the synthesized image. This work aims at overcoming these challenges by introducing different constraints to the model.

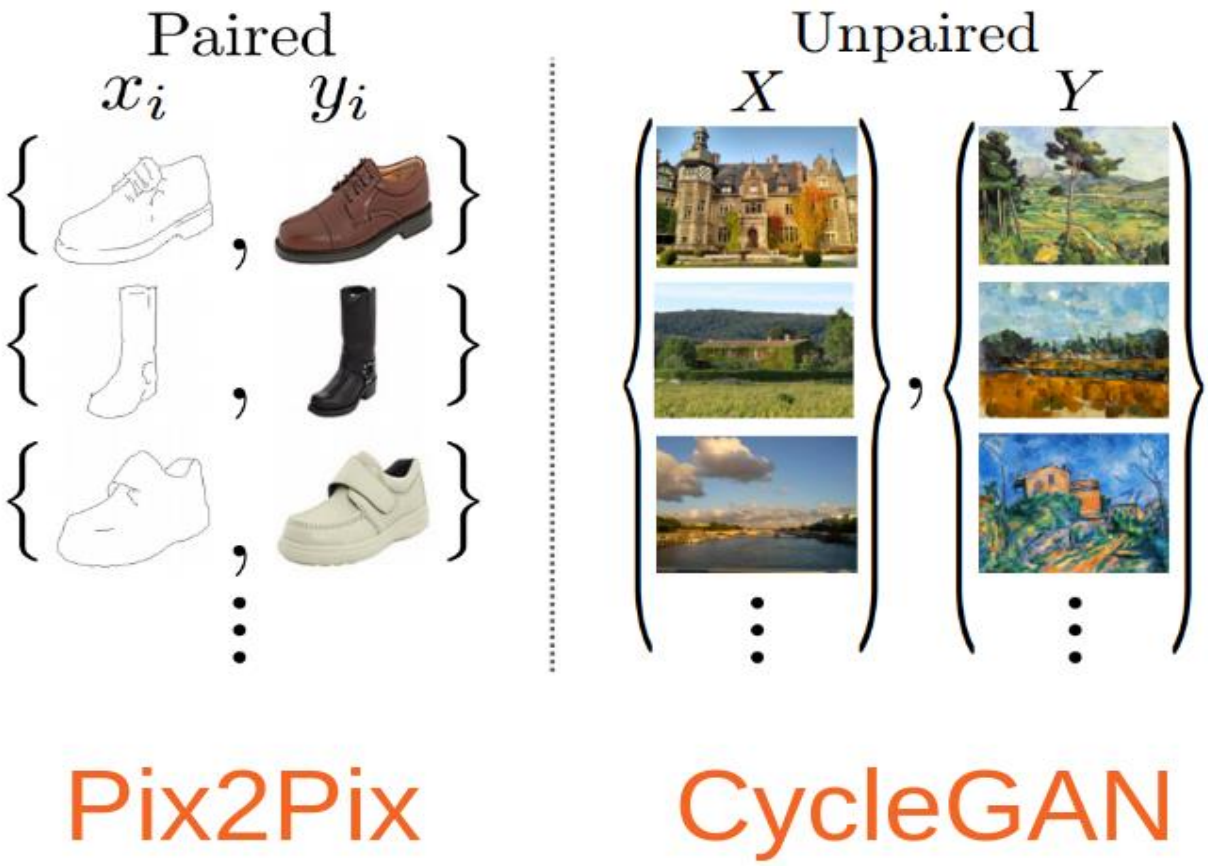


Figure 2. 5: Dataset specifications of pix2pix (paired) and CycleGAN (unpaired)

(source: cited from (Liu et al., 2017))

The other is (Choi et al., 2018) tried to overcome the problems of existing I2I limitations on providing scalability and robustness for handling multiple domains called starGAN. It is a scalable method for doing image-to-image translations across many domains using a single model. So, with a single generator, StarGAN can learn mappings across several domains.

2.6. Style transfer

The practice of moving an image's style from one domain to the other domain image is known as style transfer. Nowadays it is surprisingly incredible to see some of the recently generated images with algorithms, that look natural and real. Therefore, this style transfer is part of the image-to-image conversion, and other many works have been done since the idea of style transfer was first introduced by (L. Gatys et al., 2016). In this paper, they have done an artistic style transfer and tried to tackle the problem of style transfer using a pre-trained convolutional network for content

and style loss. (L. Gatys et al., 2016) learns to translate a single image to a single image as shown in figure 6.

Figure 2. 6: Style transfer

Where, (A) Content image. (B)-(F) are the style image and the bigger pictures are stylized images

Source: cited from (L. Gatys et al., 2016)

This style transfer made the research community excited with the results it has provided and many were working on improving the work based on the limitation it had. One of the limitations of the style transfer was that it takes all information about the style such as colors and brush strokes and transfers it to the entire content image. And this limitation was further introduced by the follow-up works of the (L. Gatys et al., 2016) authors in their paper called "Controlling Perceptual Factors in Neural Style Transfer" (L. A. Gatys et al., 2017) in which they have provided different kinds of control like, Spatial control to control the spatial position of the style transfer, color control for preserving the colors of the content in the image and, scale control for controlling the coarseness of the brush strokes. This latest work by them provided a way to create better-stylized images as shown in figure 2.8.

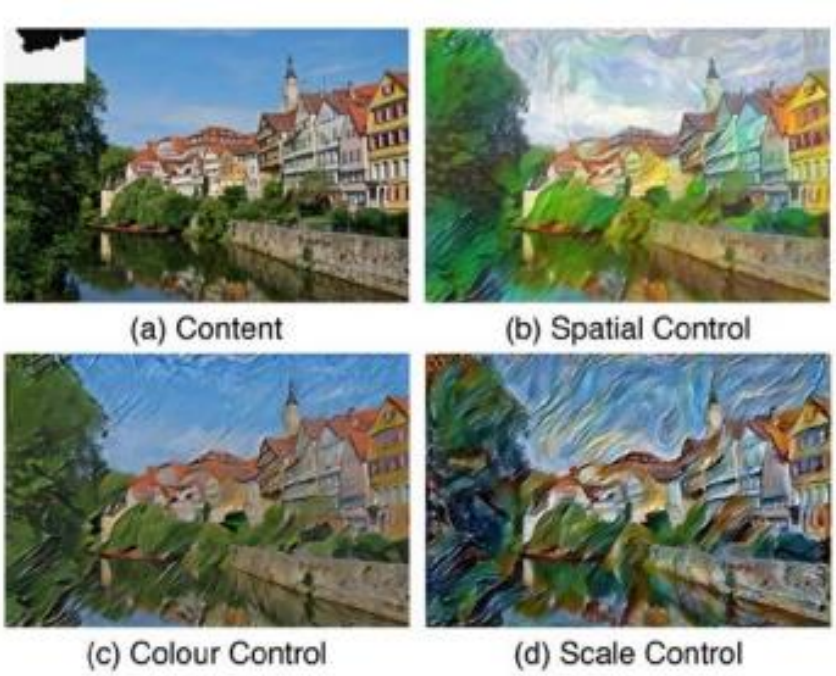


Figure 2. 7: The control methods of style transfer

(Source: cited from (L. A. Gatys et al., 2017))

2.7. Generating image of people in clothing

Nowadays generating images of people's image in clothing is having many applications and it's becoming a key area in computer vision and deep learning. Due to their application, generative modeling of people's image in clothing has been proposed in different researches. Some of the research focuses on Virtual try-on, which involves attempting to fit new in-store clothes onto a person's image via the website (online). (Sekine et al., 2014) presented a virtual fitting system that uses depth photos to estimate users' body shapes and modifies 2D clothing graphics accordingly and they are traditional methods of generating realistic images. (Pons-Moll et al., 2017) captured and retargeted clothing employing a multi-part 3D show of clothed people. (Yang et al., 2016) used a single view 2D photograph to create a 3D mesh of the outfit, which was then re-targeted to different human bodies. The problem is most of these researches rely on the 3D measurements to perform the clothing information in the generated image.

Other works are those that try to generate the images conditioned on the given poses. (De Bem et al., 2019)(Chan et al., 2019)(Esser et al., 2018)(Lassner et al., 2017) they try to generate images at

the pixel level with variants of encoder-decoder architectures. Mostly the methods they have used rely on detailed annotations of natural human images like their body pose and clothing segmentations which will be important to obtain.

Recently the task of synthesizing an image is done by deep generative models that have given flexibility in generations of the images. Especially the GAN (Ma et al., 2017)(Radford et al., 2016)(Salimans et al., 2016) and variational autoencoders (Esser et al., 2018) are being used in image generation.

2.8. Related Works

The Early works on pose-guided image generation usually focus on generating a low-resolution image or low-quality image. But recently some works show there is a capability to generate high resolution or quality images working on the particular pose and target. Some related works that are generating the images based on the pose are summarized below (Ma et al., 2017)(Balakrishnan et al., 2018)(Dong et al., 2018)(Siarohin et al., 2018)(Pumarola et al., 2018)(Uruguay, 2017) (Li et al., 2019):

(Ma et al., 2017) have tried to use the divide and conquer methods using GAN to generate the person in different poses using a two-stage process, which is first to capture the general information and synthesize the initial draft and secondly to more fill the detailed appearance problem with the introduction of GAN. For this work, the researchers have used DCGAN architecture in their implementation. But in this work, there is some limitation like on the foreground, background, and poses of the image, and further, they have tried to solve the problem on their follow up work (Ma et al., 2018) that they have disentangled the foreground, background and poses of the image into the different latent codes to make them more flexible and controllable in a divide and conquer manner. That means in this follow-up work, they have tried to overcome the limitation of their previous work by doing a two-stage work with variants of encoder-decoder architectures.

In (Siarohin et al., 2018) they have tried to introduce a deformable skip connection that can move local features on the real image to the target pose position in a UNet generator, and also they have proposed a new loss called neighbor loss instead of using L1 and L2 loss for matching the output image's details to the target image. Similarly, (Balakrishnan et al., 2018) attempted to break down

the diverse body parts into distinctive layer masks and after that applying spatial changes to each of the body parts. For the final output, the transformed segments are fused and used.

The above-discussed methods are limited to supervised settings that require a paired data during the training which is sometimes really hard to get. So, to work with the unsupervised methods or unpaired sets of data (Pumarola et al., 2018) tries to synthesize an image in different views depending on the given skeleton pose and target image by combining the help of CycleGAN, conditional adversarial, and loss functions. Generally, what they do is that they render back the generated image into the original and then apply the cycle consistency loss which will help in training the image without requiring a paired set of data.

In most current works, instead of relying on 2D key points, others choose to use 3D or 2.5D information frameworks. For example, (Uruguay, 2017) has tried to help the synthesis process by adding an estimating 3D parametric models into their framework. Similarly (Li et al., 2019) try to synthesize the 3D flow using the reference image and target pose. In (Neverova et al., 2018) have attempted to use the DensePose method from (Güler et al., 2018) to help in the warping of input textures according to their 2D coordinates and the creation of the final output in a single trainable framework from the beginning to finish. The misalignment of deep feature maps between the condition and target images is ignored in the majority of the above research and for this limitation (Dong et al., 2018) has introduced a Soft-Gated Warping-GAN with pose parser to synthesize the target parser for producing a more realistic texture for producing high resolution or quality human image conditioned on the different poses. But, still this researches lack in providing the desired results in a diversified way from a single input and also they don't give the detailed informations of the desired clothing and posture.

2.9. Summary of Related Works

The following table illustrates the summary of related works that have been described in section 2.8

Table 2. 1: Summary of related works

Papers	Architecture	Dataset	Evaluation metrics	limitation
Unpaired pose guided human image generation (Xu Chen et al., 2019).	CycleGAN and VAE	Crawled from an online fashion store and used a dataset of (Lassner et al., 2017) for pose (part model)	-Human evaluation -Nearest Neighbor Analysis (based on SSIM)	-the generated image loses the contents of the target image -there are uncommon poses that are not handled by the model - does not give any information about the background.
Pose guided human image generation (Ma et al., 2017).	DCGAN	- person re-identification Market-1501 -DeepFashion dataset	-Structural Similarity index measure (SSIM) -inception score (IS)	-has a limitation on the foreground, background, and poses in the generated image. - Does not give diversified results - Works on the paired data (requires label)
Deformable GANs for Pose-Based Human	Def-GAN	- person re-identification Market-1501	-Structural Similarity index measure (SSIM)	- need pairs of images for training the model

Image Generation (Siarohin et al., 2018)		-DeepFashion dataset	-inception score (IS) -Human evaluation	- they don't give a sample diversified images.
Unsupervised Person Image Synthesis in Arbitrary Poses (Pumarola et al., 2018).	CycleGAN, different loss functions	-DeepFashion dataset	-Structural Similarity index measure (SSIM) -inception score (IS)	-They have not considered using the background of the image in their model. - they don't give a sample diversified images.
Disentangled Person Image Generation (Ma et al., 2018)	DCGAN	-Market-1501 -DeepFashion dataset	-Structural Similarity index measure (SSIM) -inception score (IS)	-With the expense of giving the controllability the quality of the image generated has been degraded - they don't give a sample diversified images.

As shown in table 2.1 most of the researchers have tried to solve the problems of generating a human image from a 3D sketch or pose using a supervised manner which would require a paired data during training. But, sometimes getting a paired data for training is hard and expensive to obtain. Therefore, to alleviate this challenge, some researchers are trying to use unsupervised mechanisms, such as CycleGAN, which introduces a cyclic consistency method to transform from one domain to the other without paired data for training.Still, this cycle consistency lacks in providing the desired quality images and posture. The other thing is most of them work on the standard GANs discriminator which results in making training hard for the generators and unstable. So, to overcome these mentioned problems adding content preserving constraints and working on the discriminator's network would be important.

CHAPTER THREE

3. RESEARCH METHODOLOGY

3.1. Chapter Overview

This chapter tries to give a clear description of how the research methodology is done, including the data collection techniques from gathering, preprocessing to augmentation, and also the model evaluation metrics were described thoroughly.

Figure 3.1 shows how the image is gathered to its implementation. The block diagram consists of different steps from gathering, preprocessing, augmenting, and using the datasets for training and testing the model and finally evaluating the model.

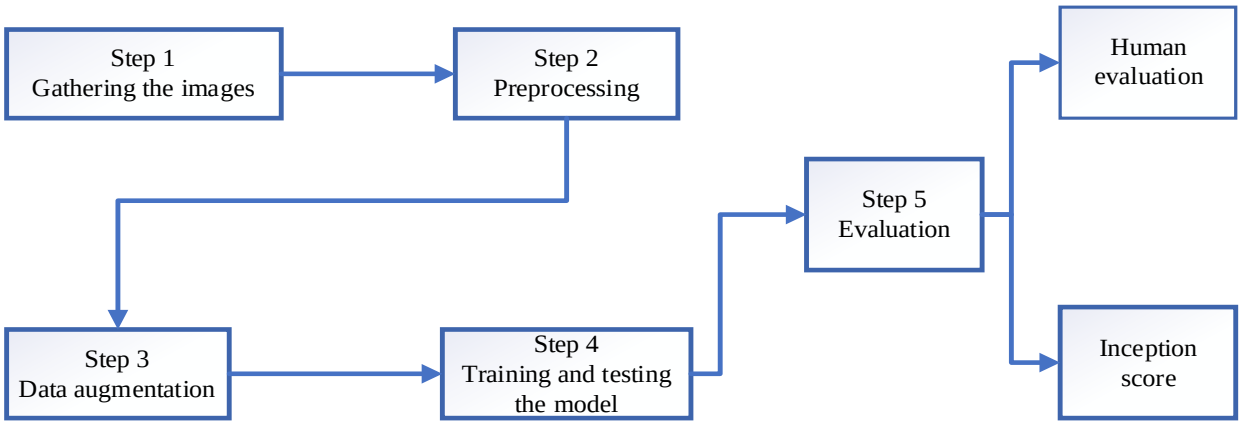


Figure 3. 1: Overall data acquisition process

3.2. Dataset Collection

The main aim of this research is to generate a human image based on a given pose or style that can work with any given image in an unsupervised manner. The dataset contains a pose and real human images and for the pose part the dataset thet were being used in (Lassner et al., 2017) were used for this research work. Also, for attaining the real human image the data's were crawled from an online fashion store[1] including the upper and full body. The datasets for the training and testing of the research are found online[2] and the link is provided below. Usually, the training of deep learning

[1] https://www.zalando.ch/
- Online fashion store where the dataset for the real image is being crawled

[2] https://drive.google.com/drive/folders/1l8nRFyXImPyDzZPNcvo9E5OyKWiT7YND #dataset location

models requires a large number of datasets for giving a better result hence, for this work data augmentation techniques are used to multiply the data. For the training, the architecture requires images of the body poses (part models) and images of real people wearing different clothes. To get more relativeness between the body parts and the real human image the dataset is prepared by separating them as full bodies (suit and dress) and Upper bodies (shirt and t-shirt).

- Full bodies (Suit and dress): this dataset is crawled from an online fashion store and it contains only images showing a full body like a suit or dress and it contains a total of 1916 real person images and 1296 pose images and total of 3212 images full bodies.
- Upper bodies (Shirt and T-shirt): it is crawled from an online fashion store and it only contains images showing upper body parts that means like a shirt or T-shirt and it contains a total of 2253 real person images and 364 pose images and total of 2617 images for upper bodies.

3.2.1. Data Augmentation

Usually, a machine learning (deep learning) model has a lot of parameters. So, when training the model, it requires a proportional amount of data to provide good results. One way to get more data, that we need is to use data augmentation, that is we need to make a slight change on the image with flipping horizontal, a shear range that is distorting of an image along an axis, and rotations with an angle of 20 to our original data. These techniques were selected because they do not affect the natural look of the dataset. So, for this work, we have used different techniques of data augmentation to make the dataset large as described in section 4.2.3. that is we have used a rotation range of 20 degrees, horizontal flip, and shear_range of 0.15 for each dataset, except for the suit and dress we have used a width shift range of 0.2, and the sample augmentation technique is shown in the figure.

Table 3. 1: Dataset size before and augmentation

Name	Before augmentation	After augmentation
Shirt and t-shirt real image	751	751*3=2253
Suit and dress real image	479	479*4=1916
Shirt and t-shirt pose image	91	91*3=364
Suit and dress pose image	324	324*4=1296

Table 3.1 describes the results of the data augmentation process before and after augmentation for the full body and upper body dataset. The augmentation produced 4 times the original dataset.

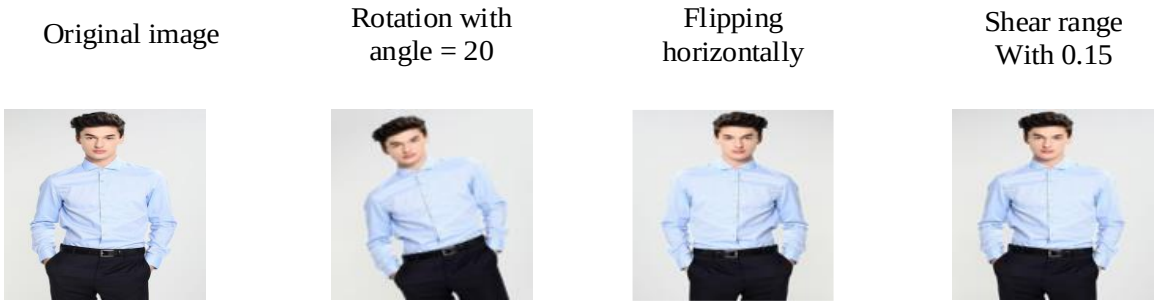


Figure 3. 2: Sample data augmentation with the different techniques used

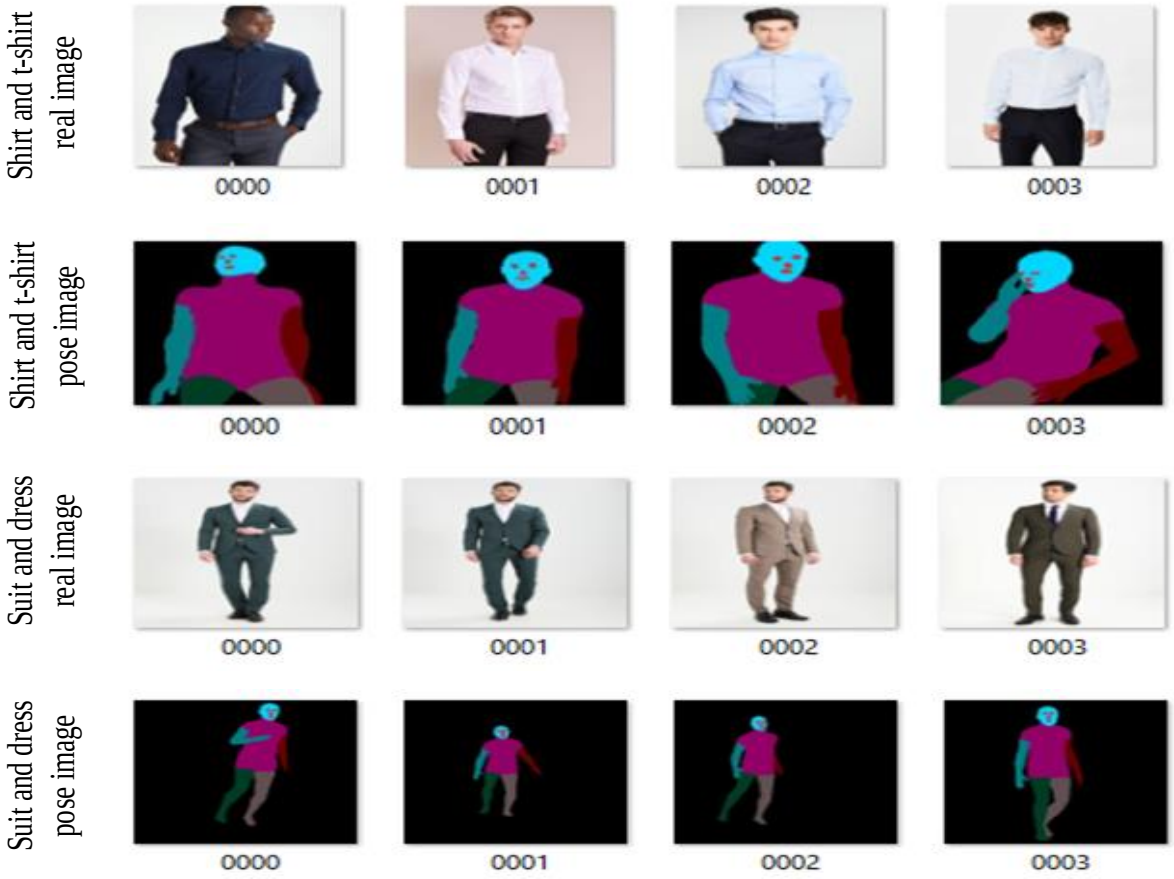


Figure 3. 3: Sample dataset images

3.3. Development Tools

For the implementation of the proposed research different software and hardware tools will be used and they are listed below.

3.3.1. Hardware Tools

The following hardware tools are going to be used in the research implementation

Table 3. 2: Hardware tools

No.	Tools	Used for
1.	RAM	For the computers speed and performance
2.	Hard Disk	For storage of large datasets
3.	GPU	For accelerating the training process

3.3.2. Software tools

For implementing the research by coding, writing different software and coding tools will be used and those are listed below with their description.

Table 3. 3: Software tools

No.	**Tools**	Specification
1.	Visio	❖ It was used to draw the proposed architecture. ❖ It is simple, easy to use tool and is compatable to use with Microsoft word easily.
2.	Anaconda	❖ It is a free and open-source distribution of python and it mainly focuses on providing everything for machine learning and data science applications. The distribution comes with a built-in Python interpreter and various packages. (*Anaconda | The World's Most Popular Data Science Platform*, n.d.) ❖ So, anaconda is used for this research because it brings many tools used in data science and machine learning with a single installation, which is very suitable for brief settings.
3.	Jupyter notebook	❖ It is an open-source web application that includes coding and real-time visualization.

		❖ It is used because it provides an easy to use mechanism with an interactive data science environment. ❖ It also, provides an easy way to combine codes,images, comments and others.
4.	Pytorch	❖ There has been a significant change in computer vision because of deep learning recently. These are because the framework and libraries of deep learning playing a great role. One of which is Pytorch, which is introduced in 2016 and hence, has much attention among the other frameworks and libraries. (*Welcome to PyTorch Tutorials — PyTorch Tutorials 1.9.0+cu102 Documentation*, n.d.). ❖ Pytorch is basically pythonic and most of researchers uses it to work on their researchs. ❖ It is easier to learn and also, it provides a Dynamic graph.

3.5. Baseline works

To verify the validity of this work, the proposed solution mainly depends on Generative models that work in an unsupervised (unpaired dataset) manner. Hence, the architecture of CycleGAN with VAE is used as a baseline work for this work. CycleGAN aims to translate an image from one domain to another (vice versa) with the help of cycle consistency loss. Therefore**,** this work aims to work on the unpaired dataset where CycleGAN is a good candidate. The VAE uses an encoder-decoder architecture in which the encoder tries to capture the information of the input into latent space and the decoder tries to regenerate the image from information in the latent space. So, for this work, the VAEs encoder could be used for capturing the detailed information of clothing to help the generator capture the details of clothing.

3.6. Feature extractor

In the formulation of this research model, the generation of clothing was performed through a variational latent space, not an image-to-image transfer. So, to get the latent space (the information about the clothing) the Variational autoencoder's Encoder without the decoder was introduced. In the encoder for the feature extraction, Resnet was used for extracting the information from the real image (y) into the latent code (z).

3.7. Evaluation Methods

Evaluation metrics are used for measuring the qualities of a model. To evaluate the performance of the method there can be objective evaluation metrics and subjective evaluation metrics. For GANs, the objective functions of generators and discriminators often measure their performance relative to their respective opponents. For example, it could be measuring how much the generator is fooling the discriminator. Some of the objective metrics that are used on GANs are Inception Score (IS) and Fréchet Inception Distance. subjective evaluation metrics like human evaluation.

3.7.1. Human Evaluation Study

To check if the generated image looks realistic human evaluation was conducted. That is the images that are generated with different experiments were compared or assessed with human evaluators. For the human evaluation, we have tried to compute the work by using a google form for those experiments explained in section 5.6 and evaluated them as to which of the results looked more realistic comparing. And for this work, the feedbacks of 20 evaluators on the google form was collected. And for the final, the average of the feedback from the evaluators is used to compute the evaluation in section 6.5.2.

3.7.2. Inception Score

It is an objective metric to evaluate the quality of the synthesized image, especially the synthetic image generated by the generative adversarial network model. (Salimans et al., 2016). The inception scoring involves image classification using a pre-trained Inception V3 deep learning neural network to classify the synthesized images (Szegedy et al., 2016). The inception score tries to measure the diversity of the image and whether the image looks like something. If both are true, the metric will be high, which means that GAN can generate many different images, as specified in (3.2).

$$IS = EXP(Ex{\sim}pgD_{\mathrm{KL}}\left(P(y|x)\middle|\middle|p(y)\right)) \tag{3.2}$$

Ideally, the generator should:

1. Generate images with significant objects
2. Generate different variety images

Higher scores are better, corresponding to a larger KL-divergence between the two distributions.

CHAPTER FOUR

4. PROPOSED ARCHITECTURE

4.1. Chapter Overview

This chapter presents the proposed solution of generating a human image based on a given pose. The generated image should have a better quality of visualization and content preservation of its original image. So, it mainly introduces; the model architecture to generate the image from pose and also the different preprocessing techniques are described thoroughly also, the different optimizing loss functions and training scheme is described.

4.2. Pre-processing Techniques

4.2.1. Normalization

Normalization is a method applied in the data preparation process. So, When the characteristics of the data have different ranges, the values of the numeric columns in the data set are changed to use the common scale of [-1, 1]. And this is calculated from the input image in the following forms.

$$output = \left(\frac{input\ image - \mu}{\delta}\right) \qquad \text{(e.q. 4.1)}$$

Where $\mu = 0.5$ and

$$\delta = 0.5$$

This way the values of the output will be in the range from -1 to 1.

4.2.2. Resize Image

The datasets have varied sets of sizes so it needs to be resized so that they could fit the input size of the network. In this work, they are resized to 128x128 following the work of (Xu Chen et al., 2019) and this resizing is important for reducing computational complexity and also to focus on the region of interest by cropping the region we are interested in.

4.2.3. Data Augmentation

The deep learning process (deep network) generally requires big datasets to provide good performance. So, in case there is limited data available data augmentation is a key solution to bring a better performance. Augmentation is a process of artificially creating training images by applying

different ways of preprocessing like; random rotation, horizontal or vertical flip, shifting, and others(*Data Augmentation - Wikipedia*, n.d.). It is a great way to expand the size of your dataset for better results.

Image Data Generator: This is a built-in Keras class that provides an easy and quick way to augment images by providing different techniques like a rotation with an angle of 20 degrees, a horizontal flip, and a shear range of 0.15.

This Image Data Generator works as the following algorithm:

Algorithm 1: Data augmentation algorithm

for all images do:

Load the original image from the disk.

Transform the images randomly using horizontal flipping, shear range=0.15, and rotation of 20-degree angle.

Write the transformed image to the disk.

end for

4.3. Proposed Aproach Architecture

The proposed architecture will mainly depend on the base CycleGAN (Zhu et al., 2017) to maintain the unsupervised scheme of the architecture with modification on the discriminator architecture and some additional loss functions like Color cycle-consistency loss and cycle complement loss that will help in preserving the contents of the generated images and improve the quality of the image. CycleGAN has introduced the cycle reconstruction loss to overcome the problem of the unpaired dataset which will be held between the real and reconstructed image. But here, as we can see there is no relationship between the generated and real image, which then results in producing less quality image and artifacts in the generated image. Therefore, to overcome this problem cycle complementary loss which will be held between the real and generated image is provided in the proposed solution. So, this loss provides an L1 loss between the real and generated image.

The other thing is the base cycle reconstruction loss encounters a problem of channel pollution which is an issue that produces artifacts on the generated image because of the influences of the different channels in the image when generating it at once. So, to overcome this problem a color

cycle consistency loss for the separate channels has been provided in this work. So, it tries to calculate the loss between each channel of the original and reconstructed image and sums up each loss at the final.

Using a standard GANs discriminator which is optimal usually results in making the training of the generator hard and unstable. Mostly these discriminators try to squeeze their output into two ends that are either 0 or 1. So to overcome this problem the relativistic average discriminator with patchGAN architecture is provided in this work. This discriminator tries to measure the difference between the generated and real images which means when it is optimal its output will be going to 0.5, that way the discriminator will have an equal odd of predicting the fake and real image. And this results in faster, better quality, and stable training of the model.

Further, the architecture of CycleGAN is improved by introducing an attention mechanism before the upsampling blocks, which will help the network in capturing the fine details from the different parts of the image and this is described in section 4.3.3. also, the VAE's encoder is used for capturing the details of clothing from the real image to the latent space in the feature extractor part of figure 4.1 and described in detail in the following section.

4.3.1. Feature Extractor

For the generation of the clothing in the real image variational latent space was used. So, to get the latent space (the information about the clothing) the Variational autoencoder's Encoder without the decoder was used as a feature extractor. So, the encoder of the VAE is trained to return the mean and covariance matrix that describes the Gaussians, which helps in expressing the latent space regularization naturally. As a result, the encoder's output distributions are generally pushed to be near to the standard normal distribution to achieve local and global regularization. Therefore, the regularization term attempts to regularize the latent space by bringing the encoder's distributions closer to a standard normal distribution (N (0,1)) and it is expressed in terms of KL-divergence between returned distribution (mean and variance) and standard Gaussian. As a result, during training, the associated KL-Divergence is reduced to regularize the latent distribution to be near to N (z), and mean (μ) and variance (σ) are encoded using the image y (real image) and mathematically it is expressed below.

$$D_{kl} = \frac{1}{2}\sum_k (E(\delta(y)^2) + \mu^2(y) - 1 - \delta(y)^2) \qquad \text{(eq.4.2)}$$

The architecture of the encoder for the feature extraction is Resnet block and it was used for extracting the information from the real image (y) into the latent code (z).

Resnet (Residual Network)

Over the last years, many researchers are trying to improve performance and solve complex problems of neural networks by adding more layers (using deep neural networks). But, it has been seen facing difficulties when adding more layers as it results in difficulties in training and also the results of the accuracy degrades over time. So, Resnet came to solve this problem by providing a residual block with the idea of introducing a skip connection. Resnet (He et al., 2016)was introduced first in 2015 and has shown great results in the ILSVRC 2015. So, the backbone in the residual block was the introduction of the skip connection which changes the output of the layers as shown mathematically in eq 4.2. and eq 4.3.

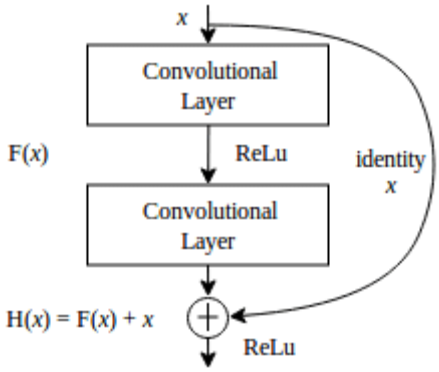


Figure 4. 1: residual block

(source: cited from (He et al., 2016))

$$H(x) = f(w_x + b) \qquad \text{(eq 4.3)}$$

$$H(x) = f(w_x + b) + x \qquad \text{(eq 4.4)}$$

Where $H(x)$= the output

$f(x)$ = the activation function

b = the bias

w_x = weight of the layer

From the above equation, we can see eq 4.2. is after the addition of the skip connection and eq 4.3 is before the addition. In general, the addition of this skip connection overcomes many problems, one of which is the problem of vanishing gradient.

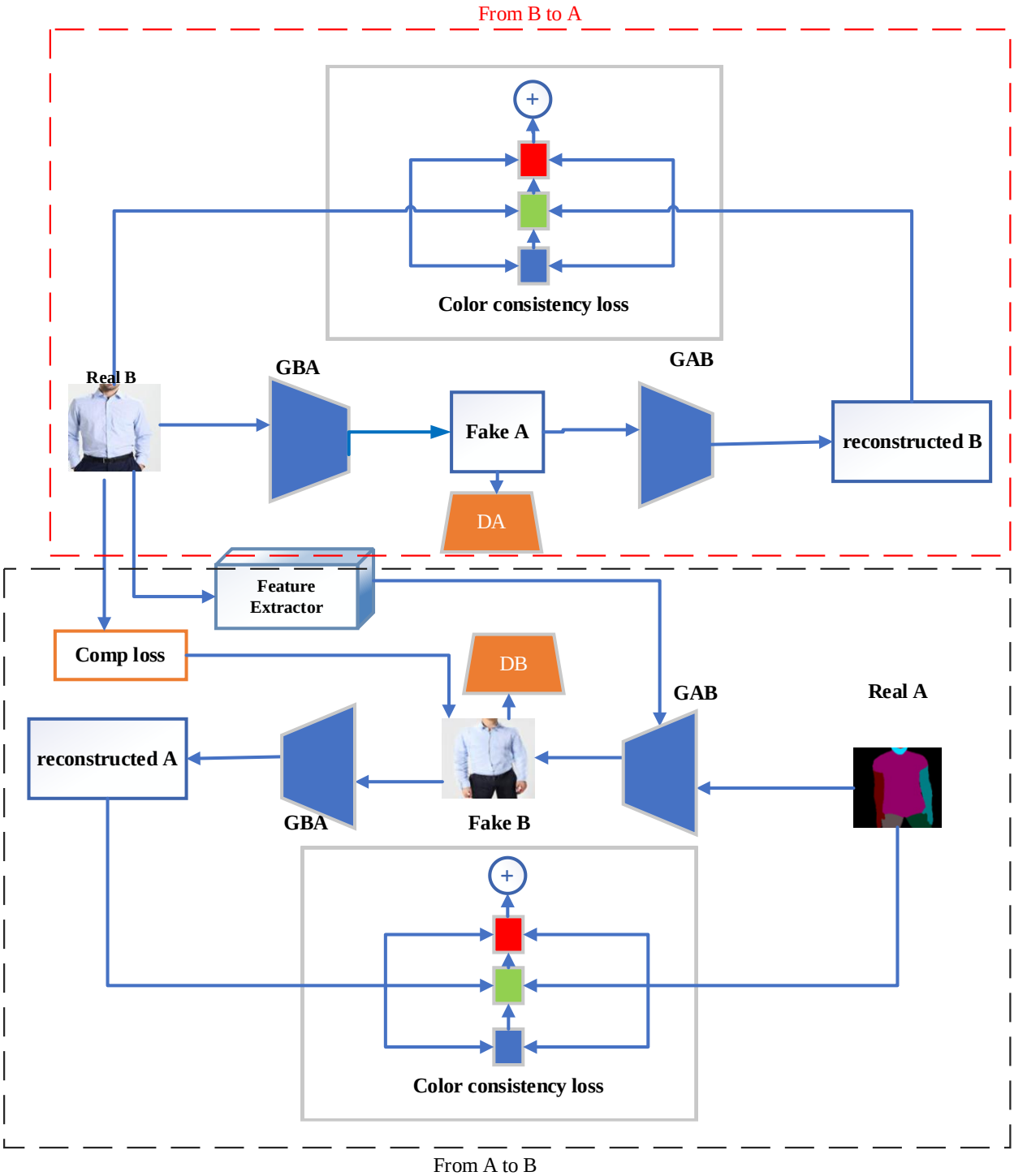


Figure 4. 2: Proposed architecture

Figure 4.2 shows the high-level model architecture of the proposed network during training that is hence the model architecture of this work mainly uses the CycleGAN architecture, at a time the generator will take an input image from one domain and translate it to the other and using the other generator it translates back to the original. But the generator A2B (GAB) takes two inputs that are the pose and the feature extracted from the original image. The other is color cycle consistency and cycle complementary were provided on the architecture. And the detailed description of the generator and discriminator network is given below.

4.3.2. Generator architecture

The generator is used for generating a realistic-looking image and it provides such performance through training, by back-propagating and updating weights in it to get the realistic synthesized image. So, the two generators in the proposed architecture of Figure 4. 2 share a common architecture that is an encoder-decoder with residual block as a bottleneck in between. As shown in figure 4.3 the generator has two down-samplings then 6 residual blocks in between followed by the attention layer and then two up-samplings followed the attention layer for each GAB and GBA. So, to describe the contents of each block:

Downsampling blocks contain:

- Convolution
- Instance normalization and
- Relu activation function.

Residual Blocks contain:

- Convolution
- Instance normalization
- Relu activation function
- Convolution and
- Instance normalization

Upsampling blocks contain:

- Transposed convolution (deconvolution)
- Instance normalization and

❖ Relu activation function

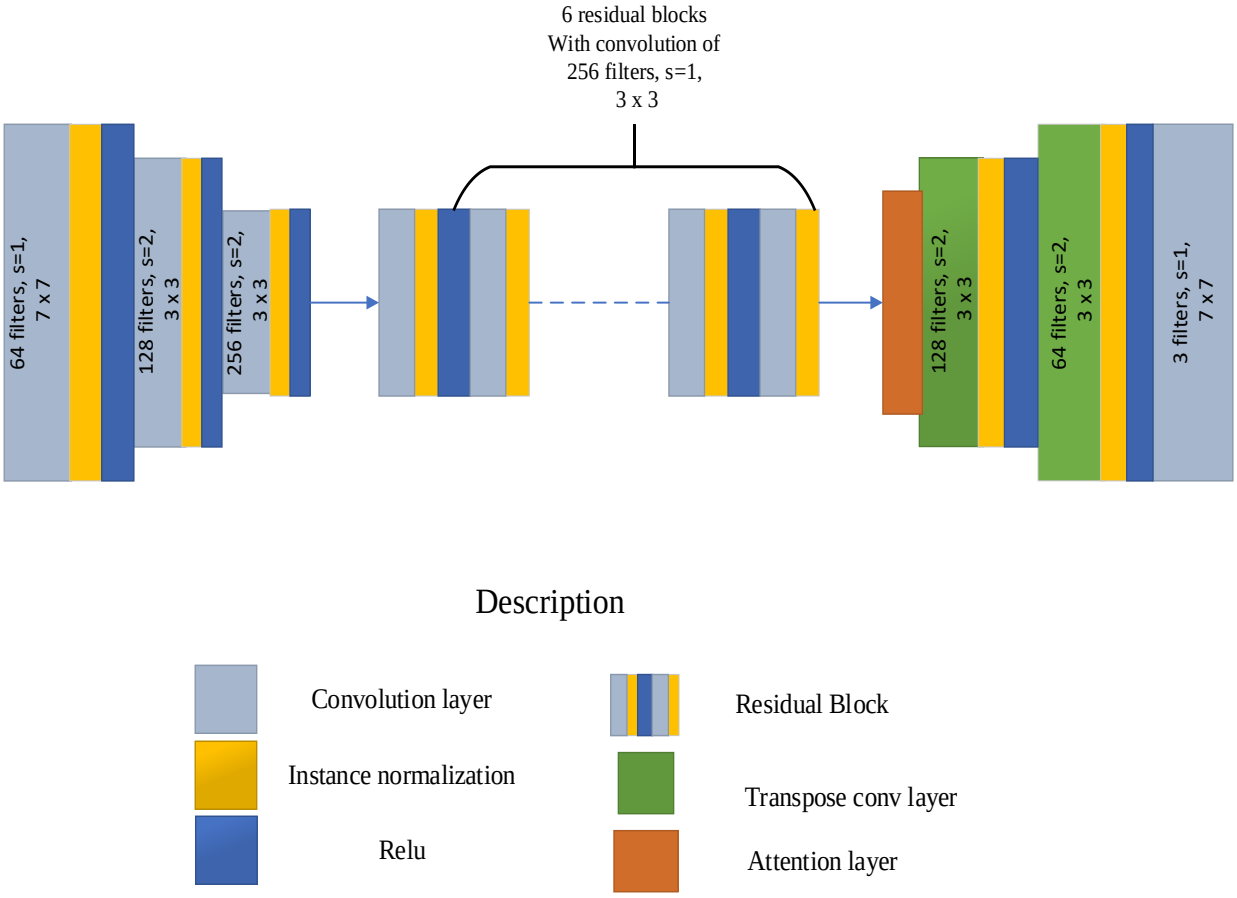


Figure 4. 3: The Generator architecture

4.4.3. Attention mechanism

Convolutional GANs usually have difficulty in learning the image distributions of multi-class datasets. That they could generate images with simpler geometry like sky easily but they fail when the images are having a specific geometry, for example, cats, dogs because convolution is a local operation that depends only on the size of the kernel. That means when computing any part of the convolution output, it has no relation with the other part except for the small local region in the size of the kernel. So to overcome this self-attention GAN (Zhang et al., 2019) has provided an attention mechanism that helps to keep the efficiency and long-range dependencies between the output of the computed convolution.

Working principle of SAGAN:

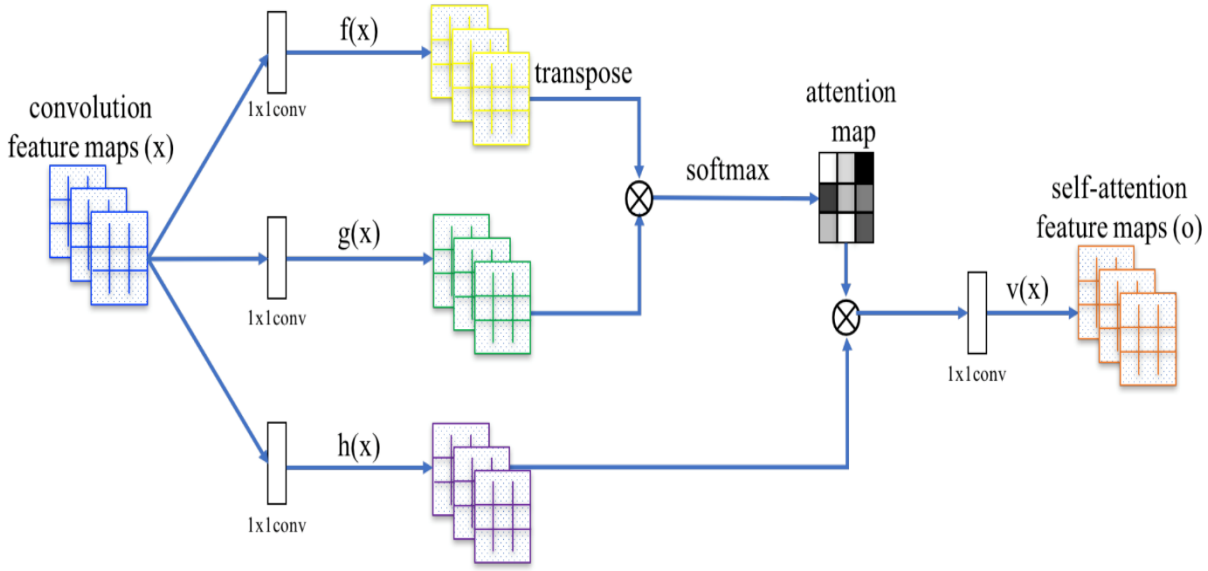


Figure 4. 4: SAGAN attention mechanism

Source ((Zhang et al., 2019))

From the figure, the feature map from a previous convolution will be further passed through three 1x1 convolutions (f, g, h) separately which will be used to reduce the number of channels in the image. The self-attention process starts after passing the feature map through the 3 convolution layers, that is the output of f is transposed and matrix multiplied with g, and softmax is taken on all rows which will produce the attention map output as shown above. Finally, the attention map output is multiplied with h to get the final self-attention feature maps. So, this layer helps a network in capturing the fine details from distant parts and it has produced great results of IS and FID (Zhang et al., 2019). So, the attention layer were used in the generators architecture of the proposed work before the upsampling block to give the detailed feature and capture the desird pose as shown in figure 4.3.

4.4.3. Relativistic Discriminator

In the standard GAN, its discriminator tries to maximize its capability to distinguish real and fake images. An optimal Discriminator usually makes training hard, which means it does not help the training of the generator or create a stable model. So, this standard GAN tries to squeeze the discriminators output into two ends that are either 0 or 1 as shown in figure 4.5 to the right.

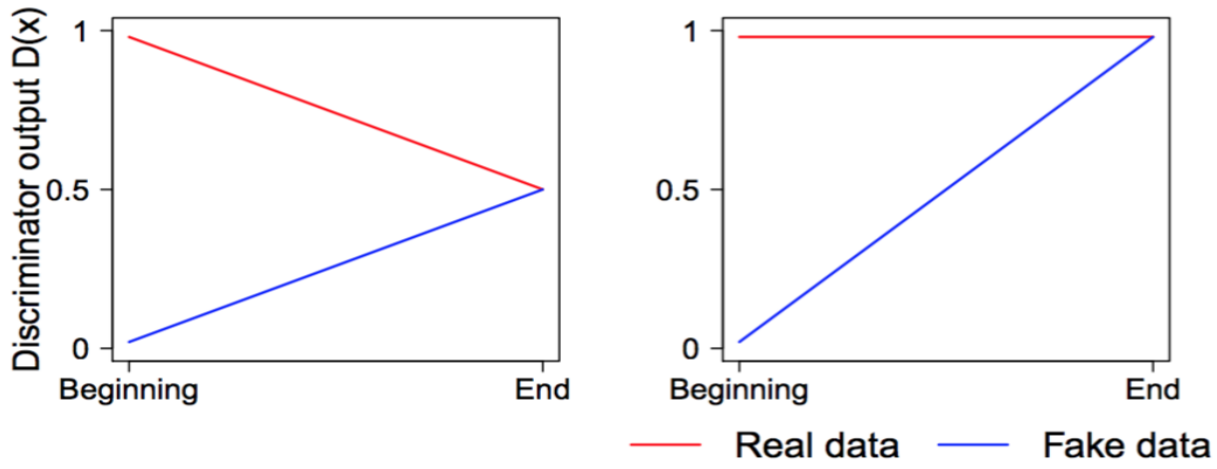


Figure 4. 5 The generator's goal with, left: Relativistic GAN and, right: standard GAN

(Source: cited from (Jolicoeur-Martineau, 2019))

Therefore, to overcome this problem of the standard GAN, the relativistic GAN is used as shown in Figure 4. 7 to the left which will try to measure the difference between the real and generated images, that means when the discriminator is optimal it will not squeeze its output to saturated areas (0 or 1) but it will make the output of the discriminator to be 0.5 that is the discriminator should have an equal odd in detecting both the real and fake images. Therefore, this solution results in more stable and faster training with a high-quality image and it is described mathematically as follows (Jolicoeur-Martineau, 2019).

SGAN (original GAN):

$$D(x) = \phi(c(x)) \qquad \text{(eq. 4.5)}$$

Relativistic GAN

$$D(x) = \phi(c(x_r) - c(x_f)) \qquad \text{(eq. 4.6)}$$

Where x is an input image (x_r for real image and x_f for fake image), ϕ is an activation function and, C(x) is an output of the discriminator.

So, the architecture of the discriminator used is a patchGAN which is shown below.

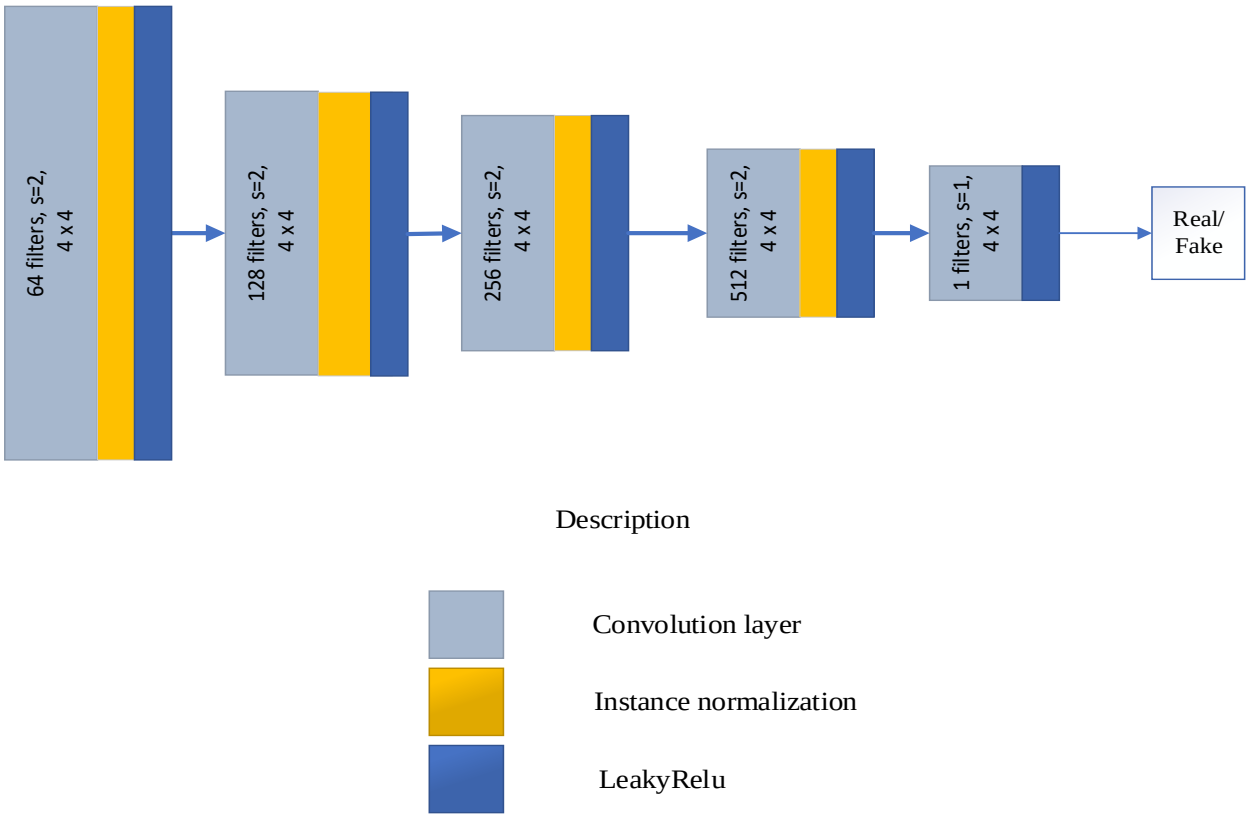


Figure 4. 6: Discriminator architecture

4.4. Model Learning Functions

To accomplish the main aim of this work different learning functions like loss function and a relativistic discriminator for a better quality generation is introduced on the base CycleGAN (Liu et al., 2017) hence the proposed work is based on the general idea of it. So, to start with the loss used in the network the standard GAN loss (ℓ_{GAN}), cyclic reconstruction loss (ℓ_{cyc}) from CycleGAN(Liu et al., 2017), latent code consistency loss (ℓ_{c}) from UPG-GAN (Xu Chen et al., 2019), complementary cycle loss (ℓ_{comp}) from (Xia et al., 2019) for better image quality and fewer artifacts, color cyclic reconstruction loss (ℓ_{ccyc}) from (Tang et al., 2019) was used to give the color consistency in generating the image and the detailed description of those are given below. So, the overall loss used in this work are as follow:

$$\ell_{overall} = \ell_{\text{GAN}} + \ell_{cyc} + \ell_{c} + \ell_{comp} + \ell_{ccyc} + \ell_{s} \qquad \text{(eq. 4.7)}$$

Cyclic reconstruction loss (ℓ_{cyc}): as described in section 2.3.3 this loss is first introduced in (Zhu et al., 2017) for overcoming the problems of paired data required for training. So, this cycle consistency loss tries to check whether the real and the twice generated (reconstructed image) are similar and it tries to minimize the mean squared distance between the two images (Y to X to Y).

$$\boldsymbol{\ell}_{cyc} = E_{x\sim p(x)}\left[\left\|G_x\left(G_y(x)\right)-x\right\|_1\right] + E_{y\sim p(y)}\left[\left\|G_y\left(G_x(y)\right)-y\right\|_1\right] \qquad \text{(eq. 4.8)}$$

Latent code consistency loss (ℓ_c): this loss is provided in this work to make the generation of the synthesized image reflect the information that is encoded in the latent space z. so the loss is represented as follows where the encoder produces the latent space.

$$\boldsymbol{\ell}_c\left(G_y, z\right) = E_{x\sim p(x), y\sim p(y)}\left[\left\|Q\left(G_y(x,z)\right) - z\right\|_1\right] \qquad \text{(eq. 4.9)}$$

Where Q is an encoder, y= real image, x= pose, z= latent space

4.4.1 Cycle compliment loss

For this work, the main target is to learn the mapping of a translation from one domain to another, so that the images from the translation are similar to that of the original image and for doing this cycle consistency loss has done a great job. So, cycle consistency loss tries to provide a mechanism to learn the mapping from one domain to another ($I_x \rightarrow I_y$) using a forward and backward consistency loss between the real and reconstructed image, but from this, there is no relationship between the real and synthesized image. So, this complementary cycle consistency loss provides an L1 loss between the real and synthesized image and then adds it to the original cycle consistency loss as shown in eq 4.7 and eq 4.8.

$$||\boldsymbol{X} - \boldsymbol{X''}||_1 + \boldsymbol{\lambda_1}||\boldsymbol{X} - \boldsymbol{X'}||_1 \qquad \text{(eq 4.10)}$$

$$||\boldsymbol{Y} - \boldsymbol{Y''}||_1 + \boldsymbol{\lambda_2}||\boldsymbol{Y} - \boldsymbol{Y'}||_1 \qquad \text{(eq 4.11)}$$

Where $\boldsymbol{X}$ is input pose, X' is generated pose, $\boldsymbol{X''}$ is reconstructed image and

$\boldsymbol{Y}$ is a real image, Y' is generated real image, and $\boldsymbol{Y''}$ is a reconstructed image

$\lambda_1$and λ_2 are weights

ss

Figure 4. 7: left: cycle consistency loss and right: complementary cycle consistency loss

4.4.2. Color cycle-consistency loss

The base CycleGAN model works on an unpaired dataset by providing a mechanism called cycle consistency loss between the real and reconstructed image to enforce the forward-backward generation, in simple terms the idea is if we translate from English to French and then from French to English, we should come to where we have started. That means when we bring to this idea, it tries to enforce getting back to the original image when translating from one domain to another and translating back to the original. This loss function is shown mathematically in eq. 4.5.

The main objective of this loss function is to minimize the difference between the original image (X) and the reconstructed image ($G_r(G_t(x, z_y), z_x)$). but with this optimization technique, we encounter a problem called "channel pollution".

The channel pollution problem is an issue that produces artifacts on an image that is generated(Tang et al., 2019). Mostly this issue arises due to the influence of the different channels in an image when generating it at once. To solve this problem, we can use a separate consistency loss for each channel and overcome the problem.

$$\boldsymbol{\ell}_{cyc}(G^t, G^r, x, z_x, z_y) = E_{x\sim p(x)}\left[\left\|G^r(G^t(x, z_y), z_x)\right\|\right] \quad \textbf{(}\text{eq 4.12})$$

$$\boldsymbol{\ell}_{cccl}(G^t, G^r, x, z_x, z_y) = \sum_{i\in\{r,g,b\}} \boldsymbol{\ell}^{\boldsymbol{i}}_{cyc} G^t, G^r, x^i, z_x, z_y \quad \textbf{(}\text{eq 4.13})$$

Where x is an input image and

x^r, x^g, x^b are the three different channels in an image

Color cycle-consistency loss: solves the "channel distribution problem" of cycle consistency loss

- The generation of an image at once makes the different channels influence each other, which then leads to artifacts.
- Instead, this proposes to construct the consistency loss for each channel separately

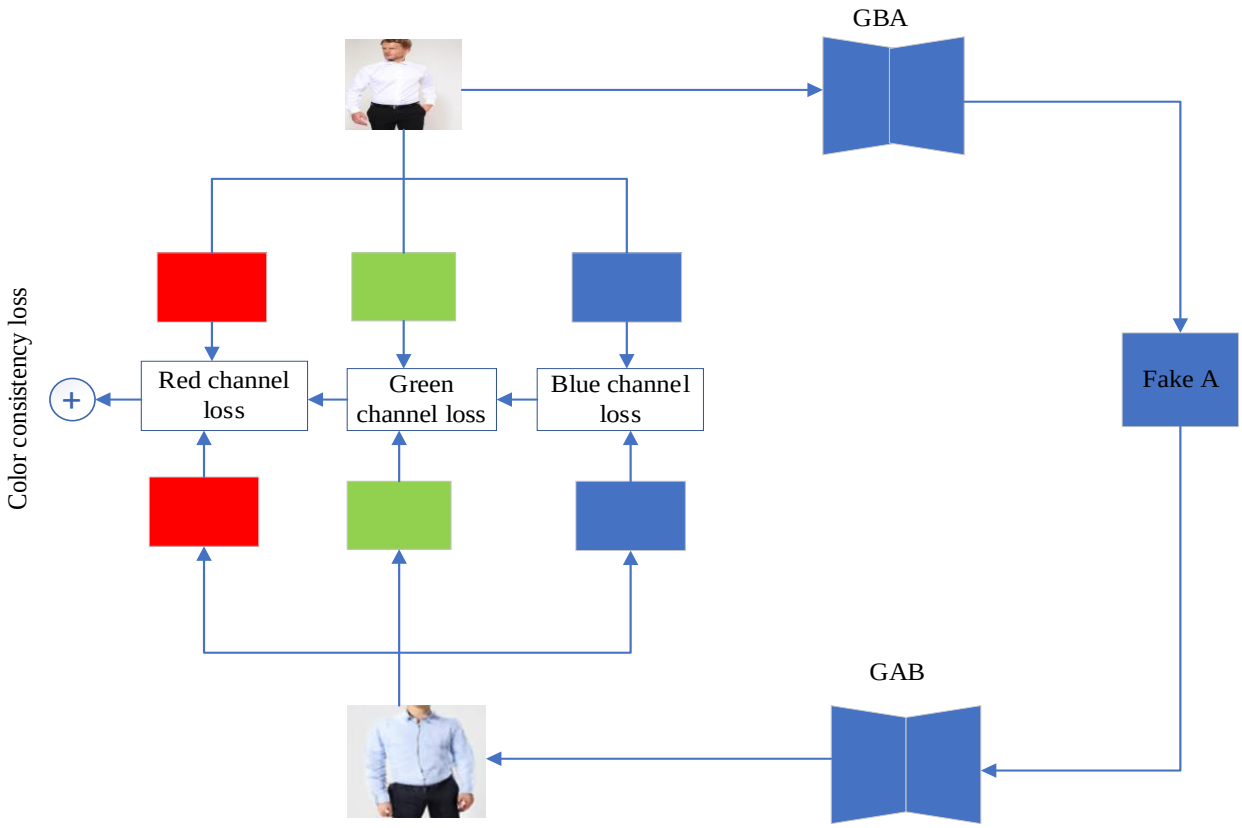


Figure 4. 8: Sample architecture showing the losses and generators

4.5. Training Pseudocode

The overall training of this work is optimized by the training pseudo-code given in algorithm 1. So, at each iteration of the training, the two inputs that are the pose and the real human image are taken from their corresponding datasets randomly. After taking the inputs the information about the human image y is encoded to the latent space z by the VAE which is then used with the input images x (pose) and y (real image) to generate synthetic images x and y. after synthesizing the real image y, the latent space z will be encoded from it. Lastly, image reconstruction is performed from the synthesized fake images and the overall loss is calculated and backpropagation through the network is performed. Below the added constaints are shown in bold.

Algorithm 2: Overall training algorithm during training

overall training algorithm of the proposed model
Input: x = pose image, y= real image, and z = latent space **For** the number of training epochs do: z=Q(y) $\hat{x} \leftarrow G_x(y)$ and $\hat{y} \leftarrow G_y(x,z)$ # synthesized image $\hat{x}$ and $\hat{y}$ $\hat{z} \leftarrow Q(\hat{y})$ #synthesized latent space from fake real image. $\tilde{x} \leftarrow G_x(\hat{y})$ and $\tilde{y} \leftarrow G_y(\hat{x},z)$ $\ell_{cyc} \leftarrow \|x - \tilde{x}\|_1 + \|y - \tilde{y}\|_1$ $\boldsymbol{\ell_{comp}} \leftarrow \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_1 + \|\boldsymbol{y} - \hat{\boldsymbol{x}}\|_1$ $\boldsymbol{\ell_{ccyc}} \leftarrow \|\boldsymbol{x^i} - \tilde{\boldsymbol{x}}^{\boldsymbol{i}}\|_1 + \|\boldsymbol{y^i} - \tilde{\boldsymbol{y}}^{\boldsymbol{i}}\|_1$ # i represents each channel in an image $\ell_c = \leftarrow \|\tilde{z} - z\|_1$ $\ell s = \leftarrow \|s - z_s\|_1$ $\theta_{D_x}, \theta_{D_y} \leftarrow \boldsymbol{Adam}(\boldsymbol{\iota}^{\boldsymbol{D}}_{\textbf{RaLSGAN}})$ $\boldsymbol{\theta_Q} \leftarrow Adam(\boldsymbol{\iota}^{\boldsymbol{D}}_{\textbf{RaLSGAN}} + \ell_{cyc} + \ell_s + D_{kl})$ $\boldsymbol{\theta_{G_x}} \leftarrow Adam(\boldsymbol{\iota}^{\boldsymbol{G}}_{\textbf{RaLSGAN}} + \ell_{cyc} + \boldsymbol{\ell_{ccyc}} + \boldsymbol{\ell_{comp}})$ $\boldsymbol{\theta_{G_y}} \leftarrow Adam(\boldsymbol{\iota}^{\boldsymbol{G}}_{\textbf{RaLSGAN}} + \ell_{cyc} + \boldsymbol{\ell_{ccyc}} + \boldsymbol{\ell_{comp}} + \ell_c)$ End **for**

CHAPTER FIVE

5. IMPLEMENTATION OF THE PROPOSED SOLUTION

5.1. Chapter Overview

Here the implementation tasks of the proposed solution are described thoroughly. Starting from environment setup, data preprocessing to the training procedures. Also, some code snippets of the implementation are explained.

5.2. Working Environments

This part describes the environment setup of the hardware used while implementing the model with their specification.

- Desktop:
  - Operating System: Windows 10
  - Processor: Intel(R) Core (TM) i7-8700 CPU @ 3.20GHz 3.19 GHz
  - Installed memory (RAM): 8.00 GB
  - System Type: 64-bit operating system, x64-based processor
- Desktop:
  - Operating System: ubuntu 20.04.2 LTS
  - Processor: Intel® Core™ i5-4590 CPU @ 3.30GHz × 4
  - Graphics: NVIDIA Corporation TU104 [GeForce RTX 2070 SUPER] / GeForce RTX 2070 SUPER/PCIe/SSE2
  - Installed memory (RAM): 8.00 GB
  - System type: 64-bit

5.3. Environment Setup

For the development of this research different kinds of software and IDEs are being used.

Anaconda: It is a free and open-source distribution of python and it mainly focuses on providing everything for data science and machine learning application.

Jupyter Notebook: It is an open-source web application that contains visualization and coding in real-time.

Colab: is a cloud-based notebook that allows writing a python code on any browser with a good computing time and resources using Tensor processing Unit or Graphical Processing Unit for training.

5.4. Training Procedure

The overall procedure to accomplish the goal of this research work, the following works have been done.

- Data gathering
- Data preprocessing
- Using data augmentation techniques to multiply the datasets
- Configuring the Graphical Processing Unit (GPU) and the dataset
- Implementing the work (from training to testing).

So, to give more information on what has been done a detailed description is provided for each phase.

5.5. Implementation Techniques for Dataset Preprocessing

5.5.1. Dataset Description

The original dataset that was available were small in number and manipulating the data to get more was necessary for this work. So, for the manipulation of data, there is a technique called data augmentation which used in this work. It tries to do the work by applying different ways of preprocessing like random rotation, shift, flip, etc. below the sample augmented images are shown.

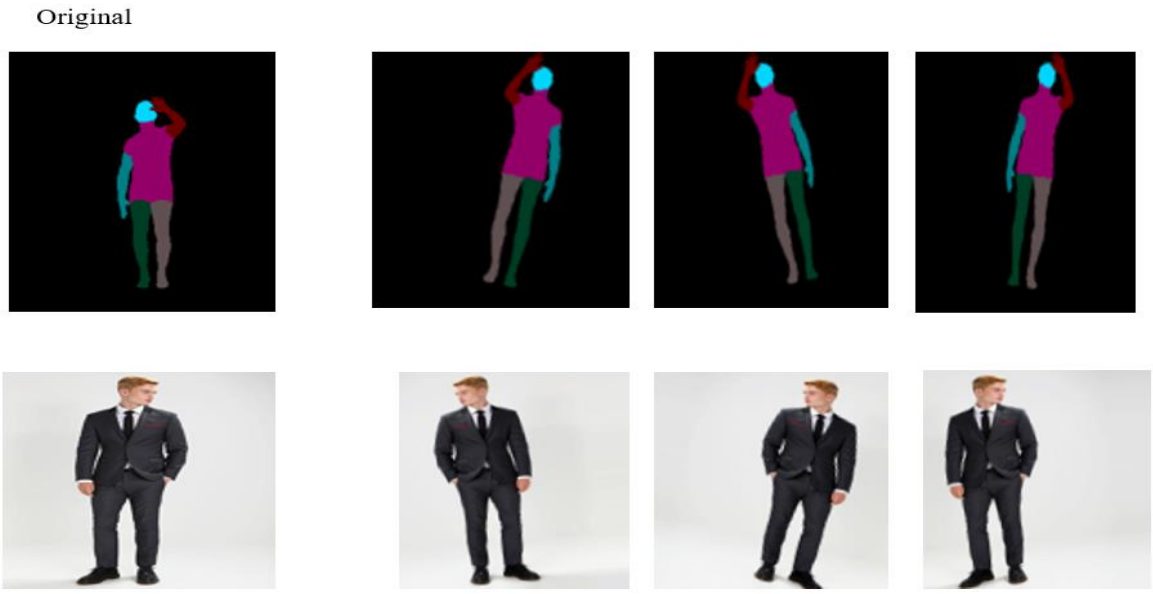


Figure 5. 1: Sample augmented dataset

5.5.2. Dataset Preparation

Image augmentation is nowadays becoming popular in deep learning as it requires a large dataset to increase the generalizability and performance of models. So, for this research work also data augmentation is used on both the pose and real human image as shown in figure 5.1. The common augmentation technique that has been used are rotation, shift and, flipping as shown below.

```
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    shear_range=0.15,
    horizontal_flip=True,
    fill_mode= "nearest"
       )
```

Code_snippet 1: Code for augmenting the datasets

5.6. CycleGAN and VAE implementations

In total, a Cycle-GAN consists of two GAN architectures: that is generator and discriminator. Generator XY: X → Y and Generator YX: Y→ X are two different variants in the generator architecture. And also, Discriminator X and Discriminator Y are specified in the discriminator architecture for discriminating the real and synthetic images. So, the layers in the architecture of the Generator and discriminators are depicted in table 5.1.

As shown in section 4.3.2. the generators take an input image with a 128x128x3 shape and try to generate an image with the same shape. So, the generator architecture consisted of a total of 12 layers that are three convolutional layers (downsampling layers) followed by six residual networks and two deconvolutional layers (upsampling layers). Also, instance normalization and Relu activation functions are used on each layer except on the last output layer. The architecture of the discriminator is almost similar to the patchGAN architecture, that is it takes an image with a shape of 128x128x3 and gives the prediction of whether the image is synthesized or real.

Code snippets for the definition of the generator and discriminator of the architecture are given as follows.

For generator

```
netG_A2B = ResnetGenerator(128,128,3)
netG_B2A = ResnetGenerator(128,128,3)
```

For discriminator

```
netD_A = Discriminator(128,128,3)
netD_B = Discriminator(128,128,3)
```

Code_snippet 2: Code for defining generator and discriminator

Table 5. 1: Generator architecture of CycleGAN

Layer	Contents of each layer
1	Convolution (filter=64, kernel = 7, stride = 1), InstanceNorm2d, Relu
2	Convolution (filter=128, kernel = 3, stride = 2), InstanceNorm2d, Relu
3	Convolution (filter=256, kernel = 3, stride = 2), InstanceNorm2d, Relu
4-9	Residual block: ❖ Convolution (filter=256, kernel =3 ,stride = 1), InstanceNorm2d, Relu
10	Deconvolution (filter=128, kernel =3 , stride = 2), InstanceNorm2d, Relu
11	Deconvolution (filter=64, kernel=3 , stride = 2), InstanceNorm2d, Relu
12	Convolution (filter=64, kernel = 7, stride = 2), InstanceNorm2d, Relu

Table 5. 2: Contents of discriminator architecture

Layer	Contents of each layer
1	Convolution (filter=64, kernel = 7, stride = 2),InstanceNorm2d Relu
2	Convolution (filter=128, kernel = 3, stride = 2),InstanceNorm2d LeakyRelu
3	Convolution (filter=256, kernel = 3, stride = 2), InstanceNorm2d, LeakyRelu
4	Convolution (filter=512, kernel = 3, stride = 2),InstanceNorm2d, LeakyRelu
5	Convolution (filter=1, kernel =3 , stride = 2),InstanceNorm2d, LeakyRelu

5.6.1. Color preserving loss

As described in the previous section 4.4.2 a color cycle-consistency loss aims to solve the problem of "channel pollution" (Tang et al., 2019) by creating the consistency loss for red, green, and blue channels separately rather than all three at once. And this way it can preserve the color losses when generating the fake image.

```
# Color loss
        self.fake_B_red = self.fake_B[:,0:1,:,:]
        self.fake_B_green = self.fake_B[:,1:2,:,:]
        self.fake_B_blue = self.fake_B[:,2:3,:,:]

        self.real_B_red = self.real_B[:,0:1,:,:]
        self.real_B_green = self.real_B[:,1:2,:,:]
        self.real_B_blue = self.real_B[:,2:3,:,:]

        self.fake_A_red = self.fake_A[:,0:1,:,:]
        self.fake_A_green = self.fake_A[:,1:2,:,:]
        self.fake_A_blue = self.fake_A[:,2:3,:,:]

        self.real_A_red = self.real_A[:,0:1,:,:]
        self.real_A_green = self.real_A[:,1:2,:,:]
        self.real_A_blue = self.real_A[:,2:3,:,:]
```

```
self.loss_G_color = (self.criterionL1(self.fake_B_red, self.real_B_red)
+   self.criterionL1(self.fake_B_green, self.real_B_green) +
self.criterionL1(self.fake_B_blue, self.real_B_blue)) *
self.opt.lambda_L1 + (self.criterionL1(self.fake_A_red, self.real_A_red)
+ self.criterionL1(self.fake_A_green, self.real_A_green) +
self.criterionL1(self.fake_A_blue, self.real_A_blue)) *
self.opt.lambda_L1
```

Code_snippet 3: Code for color consistency loss

5.6.2. Cycle compliment loss

As described in the previous section 4.4.1. this complement loss is calculated between the generated and real image rather than as cycle consistency loss between the generated and reconstructed image. So, this complementary cycle consistency loss provides an L1 loss between

the real and synthesized image and it provides an increase in quality and reduction in artifacts on generated images and code snippet showing the cycle complement loss is shown as follow:

```
# Cycle loss complement
      loss_complement_A = self.criterion_cycle(self.real_A, fake_A)
      loss_complement_B = self.criterion_cycle(self.real_B, fake_B)
```

Code_snippet 4: Code for complementary loss

5.6.3. Relativistic average discriminator

As described in section 4.4.3 relativistic discriminator will try to measure the probability that the sampled fake data is less realistic than the sampled real data (or vice versa) and it compares this measured value with the opponent's average. This way it will provide a better way for the generator to learn better. Following is the code snippet showing the relativistic average loss:

```
# Relativistic average losses
loss_real = criterion_relativivistic_GAN(real_pred_A -
fake_pred_A.mean(0, keepdim=True), valid)

loss_fake = criterion_relativivistic_GAN(fake_pred_A -
real_pred_A.mean(0, keepdim=True), fake)

# Total loss
loss_D_A = (loss_real + loss_fake) / 2
```

Code_snippet 5: Code for defining relativistic discriminator

5.7. Learning hyperparameters

For GAN to learn very well the model architecture, hyperparameters need to be carefully selected. Hyperparameters are variables that determine the network structure and network training methods. So, for this work to get a better result the following hyperparameters were used, and to describe some for optimization of the model ADAM was used. Adam is an optimization algorithm that is used to update network weight iterations based on training data and it is provided to replace the classic stochastic gradient descent process. It is used because it is simple to implement, requires small memory, is computationally efficient and it is good for problems requiring high parameters (data). The other hyperparameter is the learning rate which controls the speed at which the model learns. So, this learning rate needs to be tuned to give better results in training and this could be

done most of the time through trial and error. Here, the learning rate that is used in this work does not use a single value for the whole training, instead, it decays every 50 epochs during the training linearly from large to small and learning rate and batch size that are used as the base paper implemented.

Table 5. 3: Learning hyperparameters

No.	Hyperparameters	Value
1.	Learning rate	0.00006
2.	Batch size	1
3.	Optimizer	Adam
4.	Beta	0.5
5.	Epoch	200
6.	Weight of cycle consistency loss	10
7.	Weight of color cycle consistency loss	10
8.	Weight of KL	0.01
9.	Weight of latent code consistency (Wc)	10

Where the weights are added during training on the corresponding losses for adjusting their values.

5.8. Experiment Class

To evaluate and get the desired quality image conducting experiments and testing them are the necessary things. So, for this work, four classes of experiments were conducted. In the first class by using the base CycleGAN and VAE to generate the images based on the pose and real image given. In the second class, color-consistent cycle loss and complementary loss were added to enhance the CycleGAN and visualize the results. In the third experiment, least-square gan loss was added to the previous work and that improved the results of the work, lastly for the final work relativistic discriminator is implemented on the model. And all the experiments are summarized in the table as follows.

Table 5. 4: Lists of experimental classes

Notations	Experiment	Model
UPG-GAN	Base CycleGAN and VAE	CycleGAN
With cc + cccl	with the addition of color-preserving and complementary loss	Enhanced CycleGAN
With LSGAN	With the addition of the least square GAN	Enhanced cycleGAN
RdUPGHIG	Full implementation of my work	Enhanced CycleGAN and VAE with modification

CHAPTER SIX

6. RESULT, EVALUATION, AND DISCUSSION

6.1. Chapter Overview

Having the ideas about implementation and the details of this work in the previous sections, this chapter discusses the results obtained from the proposed solution and gives a comparative description of the results with figures, graphs, and tables.

6.2. Results of the study

6.2.1. Dataset Preparation Result

Hence the original dataset I have got was small for training I have used a data augmentation that I have tried to explain in the previous section. After the augmentation, the size of my dataset was 4 times the original dataset size. Below shows a sample augmentation of the dataset with only a horizontal flip.

Figure 6. 1: Dataset augmentation results

6.2.2. Results of the Data Augmentations

As described in the previous sections the data augmentation was used in my work for boosting the performance and results of the model and the results of the augmentation from different techniques are shown below.

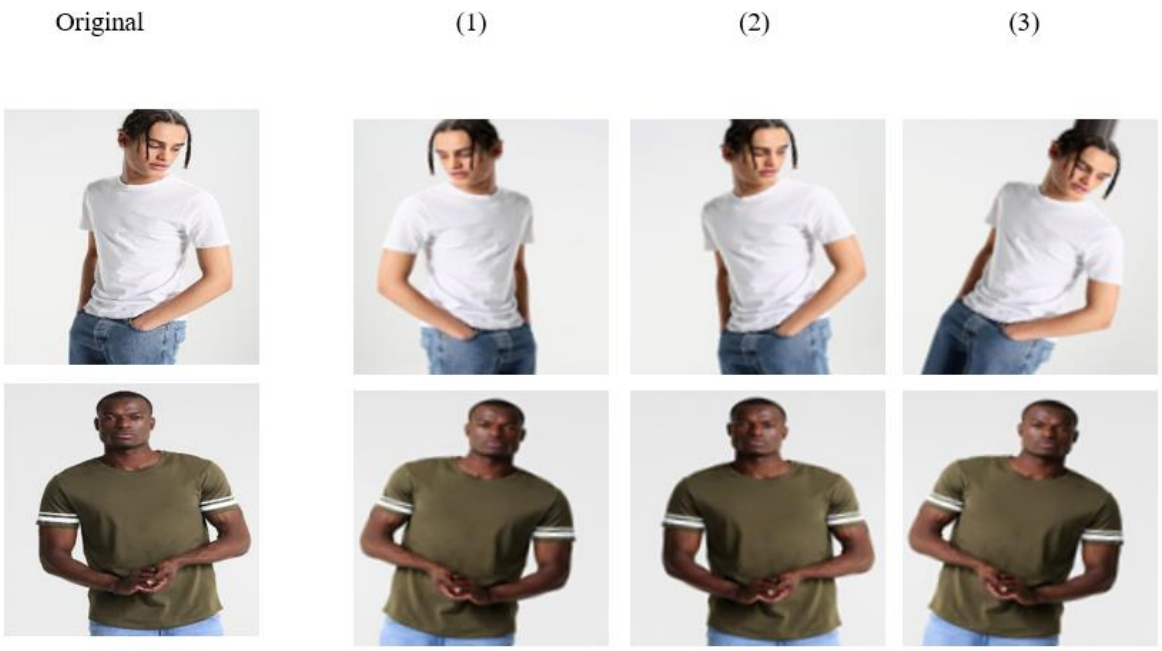


Figure 6. 2: Sample shirt and t-shirt augmented samples

Where the first column is the original image, (1) is horizontally flipped result, (2) width shift range of 0.2 results, and (3) rotation range of 20 results.

6.3. Experimental Results

For this research work, different experiments were done to get a better result and some of the experiments are as follows which are done on the same dataset, hardware, and software configurations for comparison purposes. The first one is the baseline paper's work and the second experiment was the results of adding a color-preserving (CCCL) and complementary cycle (CC) loss, the third experiment was done by adding a least-square GAN on the previous experiment and the last experiment was the final results of this research work, which means with the addition of relativistic GAN. Here, the results are shown for the two different datasets (shirt and t-shirt part and suit and dress part) for each experiment. That is the shirt and t-shirt are composed of different

images of human beings dressed shirts and t-shirt that are captured above the heist of person. The other suit and dress datasets are composed of different human images dressed in suit and dress in a full image. For all the experiments the same dataset was used for comparison.

6.3.1. UPG-GAN

This experiment shows the result of the base work on the two datasets (shirt and T-shirt datasets and suit and dress datasets)

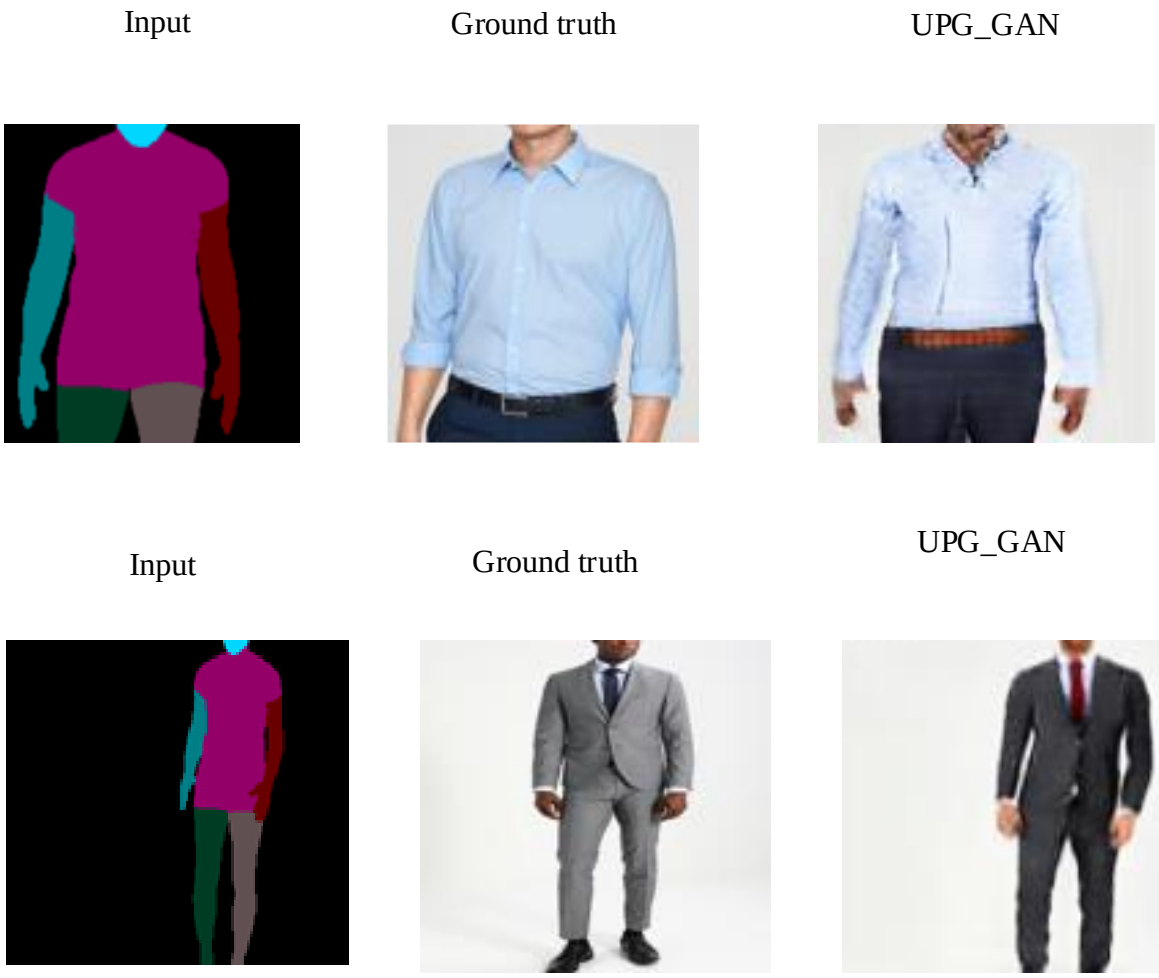


Figure 6. 3: Results of UPG_GAN

6.3.2. with cc + cccl

This result shows an improvement of the base model when adding the color-preserving loss and complementary loss. The color cycle consistency loss which is added to the model has helped the

model to capture the detailed color information of the input image on the generated image better than the previous works and also, the cycle complement which is added between the generated and ground truth has helped in providing better quality and reducing the artifacts found on the generated images and its results are shown in figure 6.4.

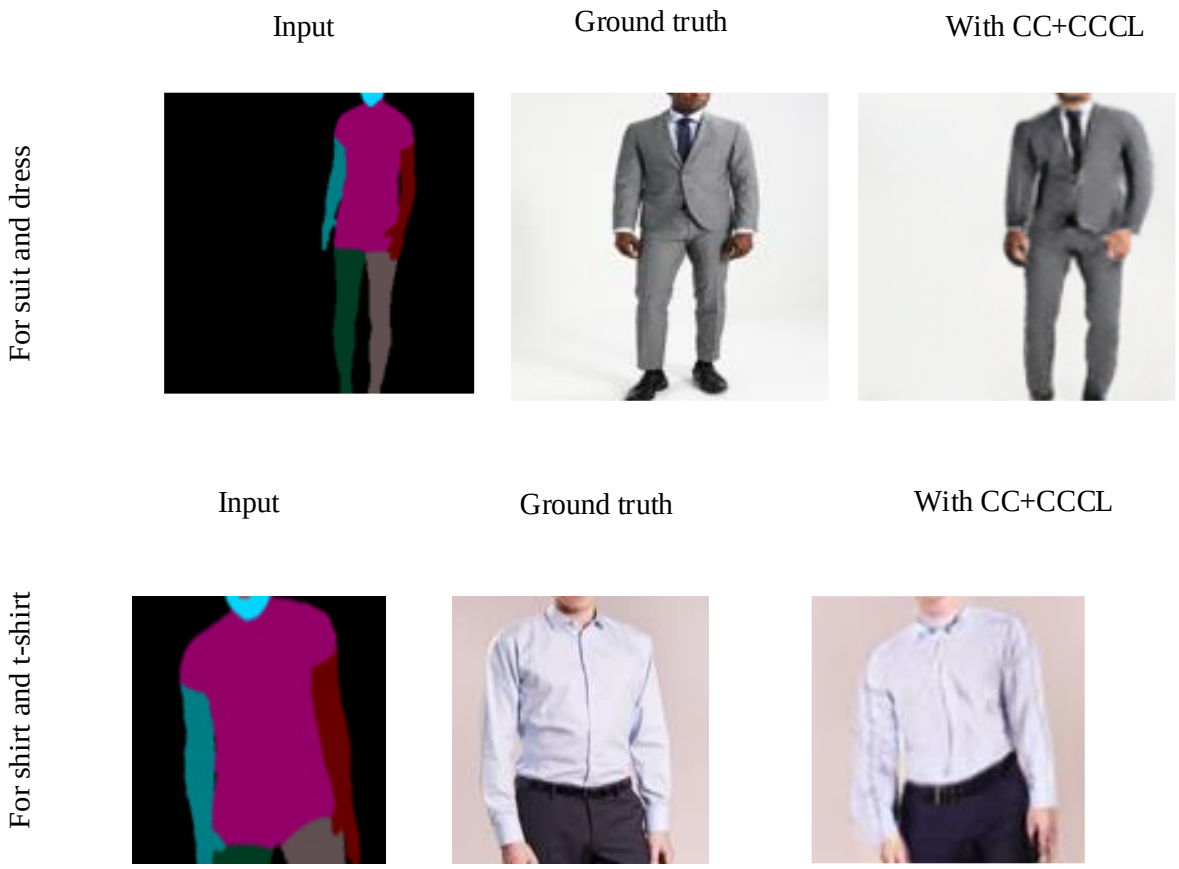


Figure 6. 4: Results with the addition of cc and cccl

Where cc is complementary cycle loss and cccl is color consistency loss

6.3.2. With LSGAN

The addition of the LSGAN to the previous work has shown an improvement in the image being synthesized. As shown below it has tried to capture the shapes and also the qualities of the image being generated have been improved. The following figure shows the results of adding least-square GAN to the previously done experiment described in section 6.3.1 above.

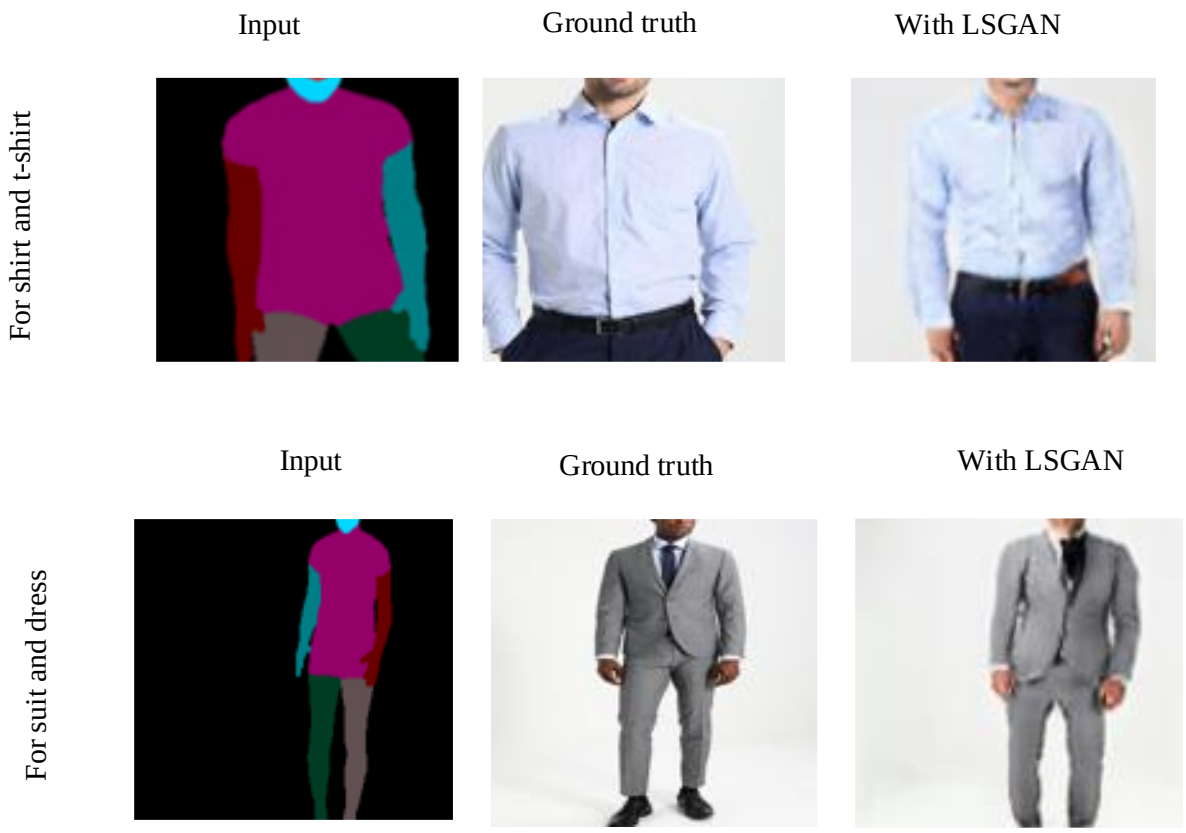


Figure 6. 5: Results of suit and dress with the addition of LSGAN

6.3.3. RD-UPGHIG

Finally, the proposed solution of this paper (Rd-UPGHIG) constitutes constraints like the color cycle consistency loss (cccl), complementary cycle loss (cc), and relativistic average discriminator. Here the addition of the relativistic average discriminator has made the network become more stable during training and helped to generate a better quality image keeping the desired pose. This is the result of the proposed solution and it gives the best result of all the experiments and is shown in figure 6.6 for the two separate datasets.

Figure 6. 6: Results of RdUPGHIG for suit

The above figure 6.7 and 6.8 shows the best result that has been obtained in this research work. And this has been further evaluated with evaluation metrics that have been introduced in section 3.7 and the results are shown in section 6.4

The other part of this study is the ability to generate a random sample image from a single pose with a variety of images that are changing the colors of the clothing in the image being generated. The results of the sample generated image from the RdUPGHIG model are shown as follows for the two datasets.

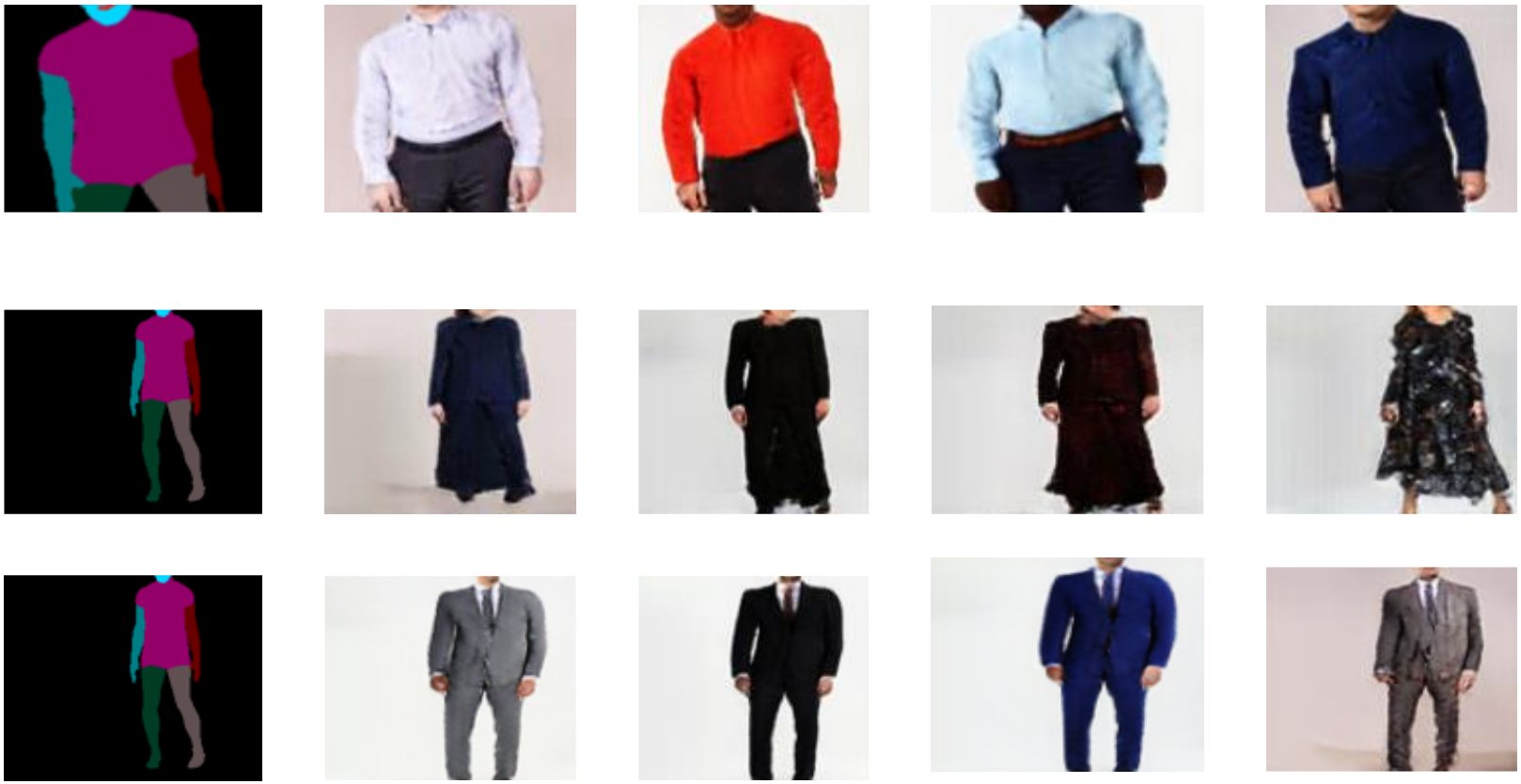


Figure 6. 7: Random image generated with RdUPGHIG

Here, the model was able to generate a random sample image of the person in a specified posture with a variety of clothing colors.

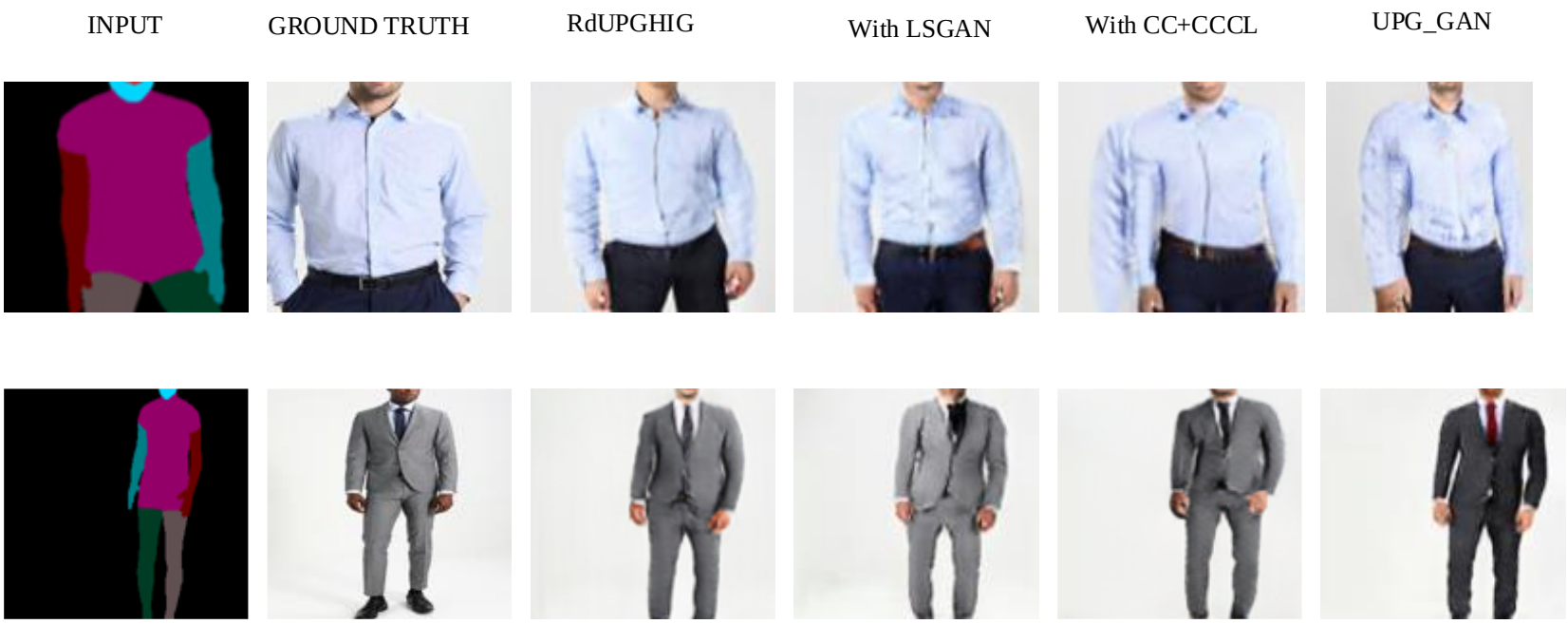


Figure 6. 8: The generation results of the different experiments (comparison and contrast)

As described above the images in this figure shows the results of the different experiments. So, here the proposed solution has provided a better visualization result, and also, it has kept the color and pose structure.

6.4. Sample training log for RD-UPGHIG

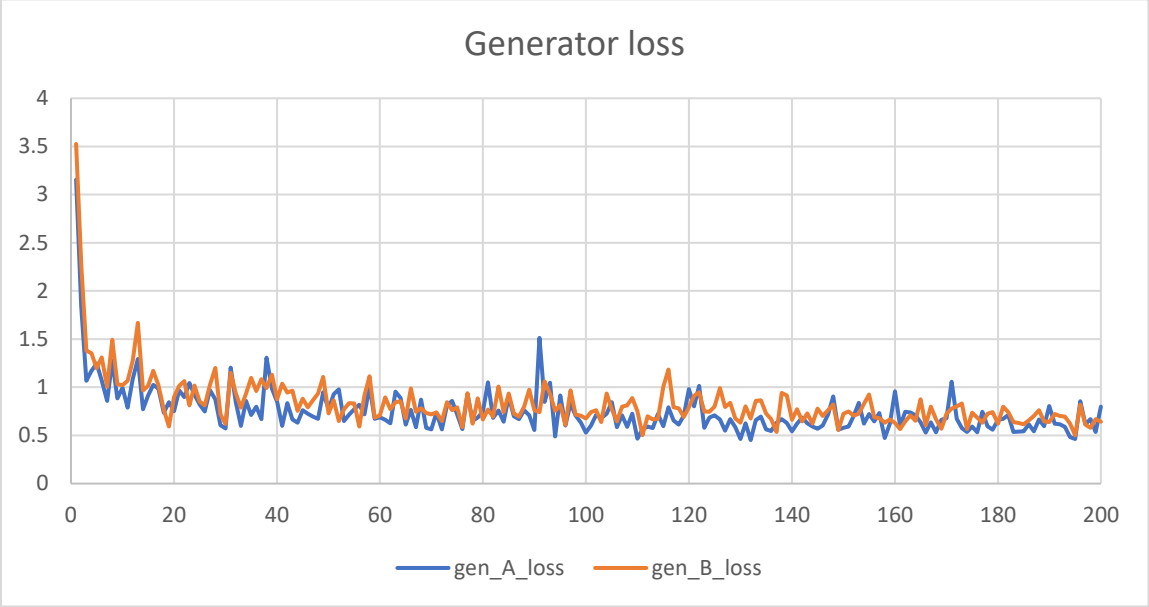


Figure 6. 9: Training loss of generator for suit and dress of RdUPGHIG

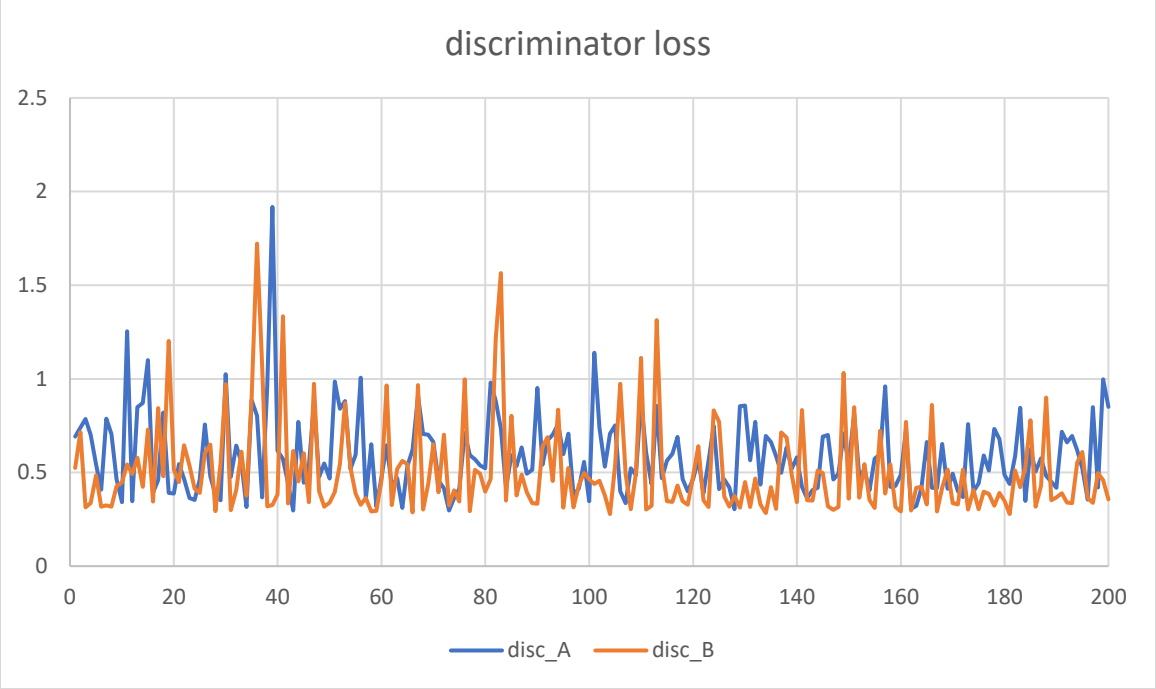


Figure 6. 10: Training loss of discriminator for suit and dress of RdUPGHIG

The two figure above shows the training losses for both the generator and discriminator (figure 6.9 and 6.10) respectively, where the generator losses are depicted for each generator (G_A with blue

and G_B with orange)) and also the losses of the discriminator are shown for both (D_A with blue and D_B with orange)

6.5. Evaluation Metrics

6.5.1. Inception Score

The graph tables below show the inception score of the different experiments. The score tries to measure the image variety and whether the image looks like something and if both are true the measure will be high and it means the GAN can generate many different distinct images and the better the model will be.

Table 6. 1: Inception score results for shirt and t-shirt and suit and dress

No.	Experiments	Inception score	
		shirt and t-shirt	suit and dress
1.	UPG-GAN	2..77 ± 0.07	2.53 ± 0.03
2.	With cc + cccl	2.84 ± 0.04	2.812 ± 0.02
3.	With LSGAN	2.9093 ± 0.03	2.9093 ± 0.05
4.	Rd-UPGHIG	**3.55 ± 0.04**	**3.01 ± 0.02**

6.5.2. Human evaluation

Here, a google form with paired images (the proposed solutions and the baseline paper's work) were prepared and the user has selected an image that looks natural and realistic capturing the desired pose. And the form was conducted as follows and the detailed overview is shown in appendix E. [3].

[3] https://docs.google.com/forms/d/e/1FAIpQLSe1agRNVoh5fe1sxFb5hI4MpYbcUjaGb8ptGa3h9BEMnUITFg/viewform?usp=sf_link

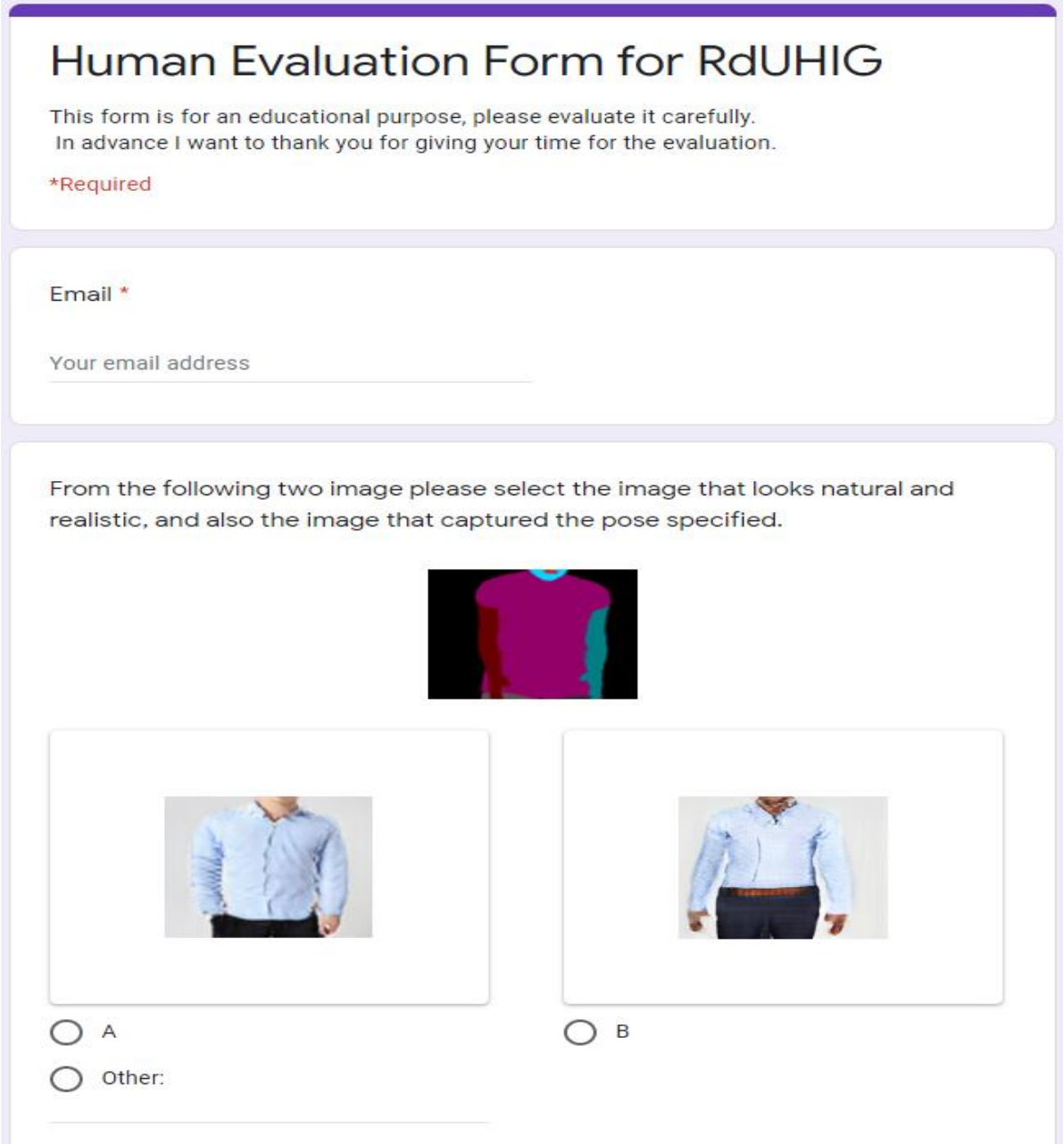


Figure 6. 11: Google form question of human evaluation

So, the average human evaluation result conducted on the twenty evaluators for shirt and t-shirt showed that the proposed solution has an improvement by an average of 50% as shown in table 6.3

Table 6. 2: Human evaluation summary on shirt and t-shirt

	Evaluation summary		
Results	Q1	Q2	Average
RdUPGHIG	13/20=0.65	17/20 = 0.85	**0.75**
UPG-GAN	7/20=0.35	3/20 = 0.15	0.25

Table 6. 3:Human evaluation results of suit and dress

	Evaluation summary			
Results	Q1	Q2	Q3	Average
RdUPGHIG	17/20=0.85	10/20=0.5	10/20=0.5	**0.6167**
UPG-GAN	3/20=0.15	10/20=0.5	10/20=0.5	0.3833

Therefore, the human evaluation study shows that the proposed solution improves the previous work by 23.34 % on the suit and dress datasets.

6.6. Discussion

6.6.1. Discussion on inception score results

To evaluate the quality and performance of the model's different metrics were used, one of which is the inception score. So, the inception score is measured for all the experiments that have been done in this work (as described in section 5.7) and their results are given in Figures 6.12 and 6.13. As we can see from the figure the result of the proposed solution is the best on both the datasets (shirt and t-shirt and suit and dress).

Therefore, the improvement on the inception score on the cc + cccl shows the improvement of the model added, which then illustrates this model helped in capturing features like the coloring of clothing and reduced the artifacts being produced on the generated images as shown in figure 6.3 and 6.4 above. Also, later on in the final proposed solution, the addition of a relativistic average discriminator made the generator capture the details and made it learn better during the training from the ground truth image making it stable.

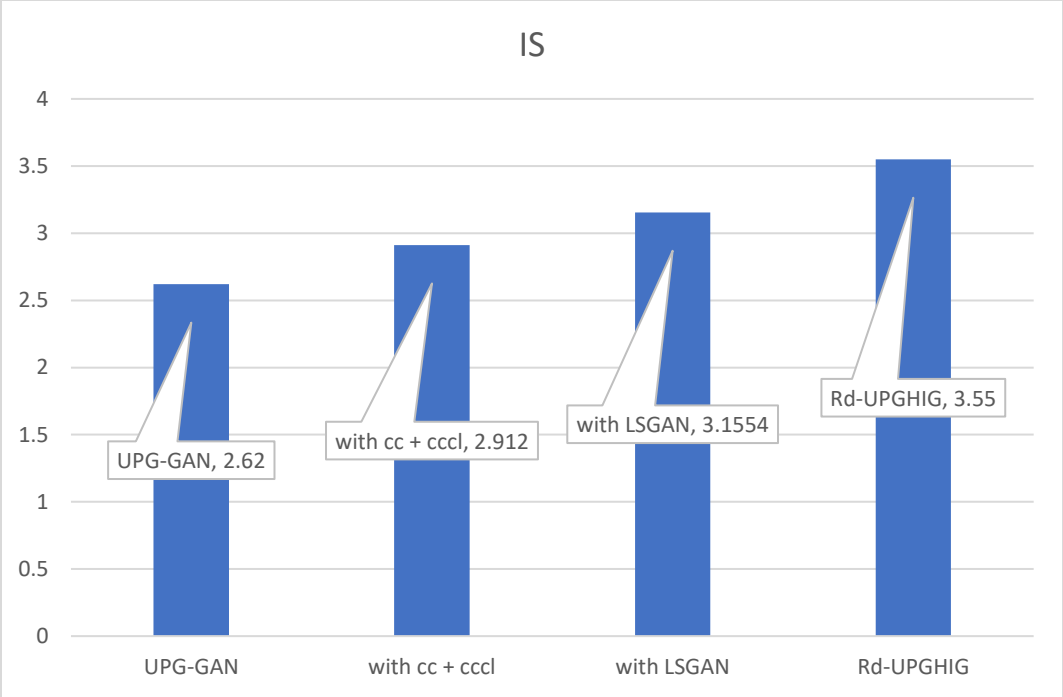


Figure 6. 12: Graph of inception score for shirt and t-shirt

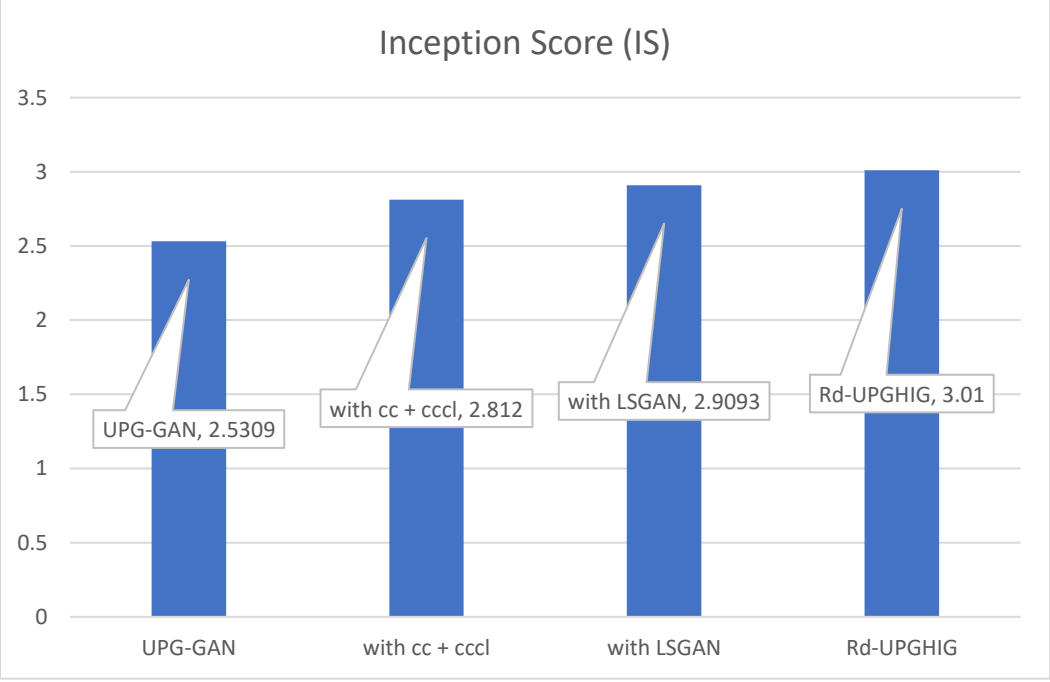


Figure 6. 13: Graph of inception score for suit and dress

6.6.2. Discussion for human evaluation

The human evaluation had given a chance for the users to give their feedback on assessing the quality of the image being produced by the model. So based on the provided paired images as described in the figure above the selected evaluators have given the results that they have thought to be real and based on their evaluation the proposed solution (Rd-UPGHIG) outperforms the baseline work. So, this evaluation took place using an average of 20 evaluators' evaluation of the images. From the result, the proposed solution improved by 50 % on the shirt and t-shirt, and 23.34 % on the suit and dress datasets as shown in table 6.3 and 6.4.

CHAPTER SEVEN

7. CONCLUSION AND FUTURE WORKS

7.1. Conclusion

Recently, there has been a great interest in generative models for image synthesis due to the capability of the images they are producing. That means being able to generate a photo-realistic image given that a pose or sketch of the person haves many applications in different areas like fashion and the E-Commerce industry, image editing, synthesizing training data, and many others but it is sometimes hard to get the training data for producing such application in a supervised (paired data's) scenario, and also, the images being generated have a problem in depicting the specified pose, color consistency and the overall images being generated have an unwanted artifact. So, this work was aimed to improve those problems by providing different constraints to the base CycleGAN and VAE networks trained on unpaired datasets.

This work introduces the way of generating a human image in clothes based on generative modeling that is taking only a pose as an input, it generates realistic and natural-looking images that represent the given input pose. Also, it gives controllability of the specified color and can give a sample of varied images with different colors for a single pose given. To do so, this research work has provided a color cycle consistency loss, complementary loss, and relativistic average discriminator. These additions to the CycleGAN have indeed shown improvements in the results being generated as shown in section 6.3.

Overall, in this study, three different experiments were done for comparison. The first experiment was the effect of adding the color-preserving loss and complementary loss and it has shown an improvement in preserving the color and reducing the generation of unwanted artifacts on the synthesized images. The second experiments were the effect of adding a least-square GAN to the model and it has shown improvements in the qualities of the image being generated than the first experiment as it adds stability to the training of the generator. The other experiment was the final proposed solution and it adds a relativistic discriminator to the first experiment which gives more stability to the network and also provides higher quality images compared to the early works that use SGAN for the discriminator that makes both the generator and discriminator optimal

(overfitted) which in turn makes training hard. So, using the proposed solution gave regularization to the discriminator to overcome the problem.

The RdUPGHIG (proposed solution) has shown that great improvements with the addition of the different constraints to the learning function that it has been trained on. The model was evaluated with an inception score and it has given a better result than the other comparative models as shown in section 6.5.1. also, a human evaluation study has shown that the proposed solution gives a better result and performance by improving the base work by 50 % on upper body and 23.34% on the full body.

Finally, this research work concludes that the addition of a relativistic average discriminator has indeed improved the stability and the quality of the image being generated, and further the addition of color-preserving and complementary loss has shown a better image color consistency and reduced unwanted artifacts.

7.2. Future Work

Even though, this research has been experimented with the fundamental deep learning model in unsupervised and generative modeling. It is limited in providing detailed information about the background of the image. So, the generative modeling can be further conditioned with different patterns to produce the desired background information that means the information on the background and foreground could be disentangled and processed individually. The other important thing in this experiment was the feature extractor the relativistic discriminator used in the experiment because they capture the detailed information of the real image, help the generator training, and quality image. So this can further be improved by providing a mechanism to use a better feature extractor with other conditions and also introducing a multiscale relativistic discriminator which will try to discriminate an image at different scales to more improve the qualities. Further it will be great to take this work to a temporal (video generation) with a modification to the network for handling the consistency of frames with desired pose. Finally, to capture the different failed pose styles it could be great to add more datasets with diversified styles and clothing which will lead to producing great results.

References

Anaconda | The World's Most Popular Data Science Platform. (n.d.). Retrieved July 26, 2021, from https://www.anaconda.com/?__cf_chl_captcha_tk__=pmd_a384644287a87d01d3776ff7dc8361f82c448a86-1627290371-0-gqNtZGzNArijcnBszQu6

Balakrishnan, G., Zhao, A., Dalca, A. V., Durand, F., & Guttag, J. (2018). Synthesizing Images of Humans in Unseen Poses. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 8340–8348. https://doi.org/10.1109/CVPR.2018.00870

Chan, C., Ginosar, S., Zhou, T., & Efros, A. (2019). Everybody dance now. *Proceedings of the IEEE International Conference on Computer Vision*, *2019-Octob*, 5932–5941. https://doi.org/10.1109/ICCV.2019.00603

Chen, Xi, Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., & Abbeel, P. (2016). InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in Neural Information Processing Systems*, 2180–2188.

Chen, Xu, Song, J., & Hilliges, O. (2019). Unpaired pose guided human image generation. *ArXiv*.

Cheong, S. Y. (n.d.). *Hands-On Image Generation with TensorFlow*.

Choi, Y., Choi, M., Kim, M., Ha, J. W., Kim, S., & Choo, J. (2018). StarGAN: Unified Generative Adversarial Networks for Multi-domain Image-to-Image Translation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 8789–8797. https://doi.org/10.1109/CVPR.2018.00916

Data augmentation - Wikipedia. (n.d.). Retrieved July 31, 2021, from https://en.wikipedia.org/wiki/Data_augmentation#Transformations_of_images

De Bem, R., Ghosh, A., Boukhayma, A., Ajanthan, T., Siddharth, N., & Torr, P. (2019). A conditional deep generative model of people in natural images. *Proceedings - 2019 IEEE Winter Conference on Applications of Computer Vision, WACV 2019*, 1449–1458. https://doi.org/10.1109/WACV.2019.00159

Dong, H., Liang, X., Gong, K., Lai, H., Zhu, J., & Yin, J. (2018). Soft-gated warping-GaN for pose-guided person image synthesis. *Advances in Neural Information Processing Systems*, *2018-Decem*, 474–484.

Esser, P., Sutter, E., & Ommer, B. (2018). A Variational U-Net for Conditional Appearance and Shape Generation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 8857–8866. https://doi.org/10.1109/CVPR.2018.00923

Gatys, L. A., Ecker, A. S., Bethge, M., Hertzmann, A., & Shechtman, E. (2017). Controlling perceptual factors in neural style transfer. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, *2017-Janua*(c), 3730–3738. https://doi.org/10.1109/CVPR.2017.397

Gatys, L., Ecker, A., & Bethge, M. (2016). A Neural Algorithm of Artistic Style. *Journal of Vision*, *16*(12), 326. https://doi.org/10.1167/16.12.326

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, *63*(11), 139–144. https://doi.org/10.1145/3422622

Güler, R. A., Neverova, N., & Kokkinos, I. (2018). DensePose: Dense Human Pose Estimation in the Wild. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 7297–7306. https://doi.org/10.1109/CVPR.2018.00762

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, *2017-Janua*, 5967–5976. https://doi.org/10.1109/CVPR.2017.632

Jolicoeur-Martineau, A. (2019). The relativistic discriminator: A key element missing from standard GaN. *7th International Conference on Learning Representations, ICLR 2019*.

Kingma, D. P., & Welling, M. (2014). Auto-encoding variational bayes. *2nd International*

Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings, *Ml*, 1–14.

Lassner, C., Pons-Moll, G., & Gehler, P. V. (2017). A Generative Model of People in Clothing. *Proceedings of the IEEE International Conference on Computer Vision*, *2017-Octob*, 853–862. https://doi.org/10.1109/ICCV.2017.98

Lecun, Y., Bottou, L., Bengio, Y., & Ha, P. (1998). LeNet. *Proceedings of the IEEE*, *November*, 1–46.

Li, Y., Huang, C., & Loy, C. C. (2019). Dense intrinsic appearance flow for human pose transfer. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2019-June*, 3688–3697. https://doi.org/10.1109/CVPR.2019.00381

Liu, M. Y., Breuel, T., & Kautz, J. (2017). Unsupervised image-to-image translation networks. *Advances in Neural Information Processing Systems*, *2017-Decem*(Nips), 701–709.

Ma, L., Jia, X., Sun, Q., Schiele, B., Tuytelaars, T., & Van Gool, L. (2017). Pose guided person image generation. *Advances in Neural Information Processing Systems*, *2017-Decem*(Nips), 406–416.

Ma, L., Sun, Q., Georgoulis, S., Van Gool, L., Schiele, B., & Fritz, M. (2018). Disentangled Person Image Generation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 99–108. https://doi.org/10.1109/CVPR.2018.00018

Mirza, M., & Osindero, S. (2014). *Conditional Generative Adversarial Nets*. 1–7. http://arxiv.org/abs/1411.1784

Neverova, N., Alp Güler, R., & Kokkinos, I. (2018). Dense Pose Transfer. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11207 LNCS*, 128–143. https://doi.org/10.1007/978-3-030-01219-9_8

Pons-Moll, G., Pujades, S., Hu, S., & Black, M. J. (2017). ClothCap: Seamless 4D clothing capture and retargeting. *ACM Transactions on Graphics*, *36*(4). https://doi.org/10.1145/3072959.3073711

Pumarola, A., Agudo, A., Sanfeliu, A., & Moreno-Noguer, F. (2018). Unsupervised Person Image Synthesis in Arbitrary Poses. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 8620–8628. https://doi.org/10.1109/CVPR.2018.00899

Radford, A., Metz, L., & Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 1–16.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (2013). Learning Internal Representations by Error Propagation. *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*, *V*, 399–421. https://doi.org/10.1016/B978-1-4832-1446-7.50035-2

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training GANs. *Advances in Neural Information Processing Systems*, 2234–2242.

Sekine, M., Sugita, K., Perbet, F., Stenger, B., & Nishiyama, M. (2014). *Virtual Fitting by Single-Shot Body Shape Estimation*. *October 2014*, 406–413. https://doi.org/10.15221/14.406

Siarohin, A., Sangineto, E., Lathuiliere, S., & Sebe, N. (2018). Deformable GANs for Pose-Based Human Image Generation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 3408–3416. https://doi.org/10.1109/CVPR.2018.00359

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 2818–2826. https://doi.org/10.1109/CVPR.2016.308

Tang, H., Xu, D., Wang, W., Yan, Y., & Sebe, N. (2019). Dual Generator Generative Adversarial Networks for Multi-domain Image-to-Image Translation. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11361 LNCS*, 3–21. https://doi.org/10.1007/978-3-030-20887-5_1

Uruguay, I. (2017). Manal de producción de semilla de raigrás anual. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 5391–5399.

Welcome to PyTorch Tutorials — PyTorch Tutorials 1.9.0+cu102 documentation. (n.d.). Retrieved July 26, 2021, from https://pytorch.org/tutorials/

Welling, M. (2017). *Variational Auto-Encoders*. https://www.jeremyjordan.me/content/images/2018/06/Screen-Shot-2018-06-20-at-2.47.56-PM.png

Xia, W., Yang, Y., & Bao, X. yu. (2019). ECGAN: Image translation with multi-scale relativistic average discriminator. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 11516 LNCS* (Issue July). Springer International Publishing. https://doi.org/10.1007/978-3-030-23367-9_3

Yang, S., Ambert, T., Pan, Z., Wang, K., Yu, L., Berg, T., & Lin, M. C. (2016). *Detailed Garment Recovery from a Single-View Image*. *VV*. https://doi.org/10.1145/XXXXXXX.YYYYYYY

Zhang, H., Goodfellow, I., Metaxas, D., & Odena, A. (2019). Self-attention generative adversarial networks. *36th International Conference on Machine Learning, ICML 2019*, *2019-June*(May), 12744–12753.

Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. *Proceedings of the IEEE International Conference on Computer Vision*, *2017-Octob*, 2242–2251. https://doi.org/10.1109/ICCV.2017.244

Appendixes

Appendix A: Sample of the datasets

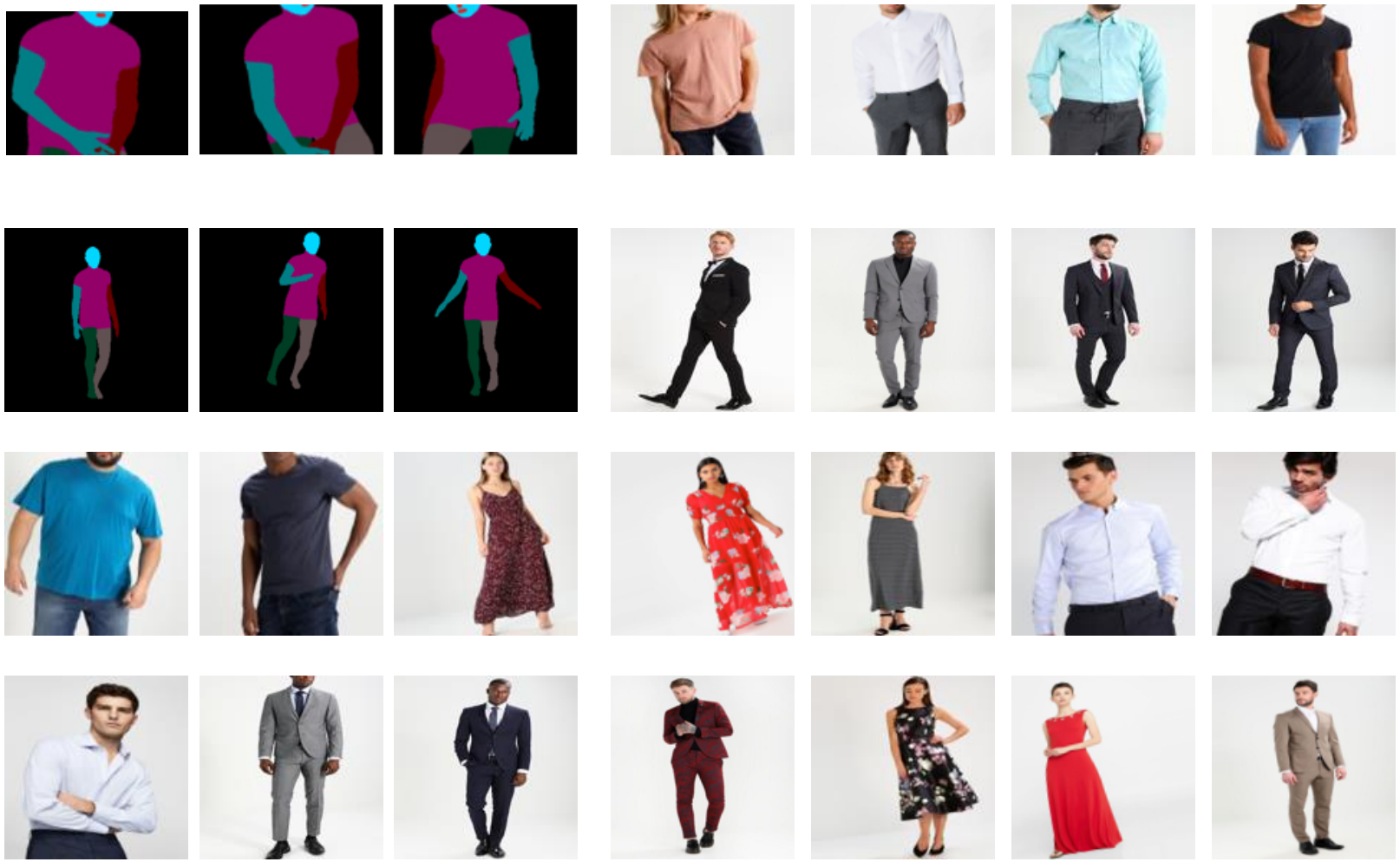


Figure A. 1: Sample datasets

Appendix B: Training log curves

The training losses of the generator and the discriminator of each experiment were as follows. All the experiments have been trained for 200 epochs.

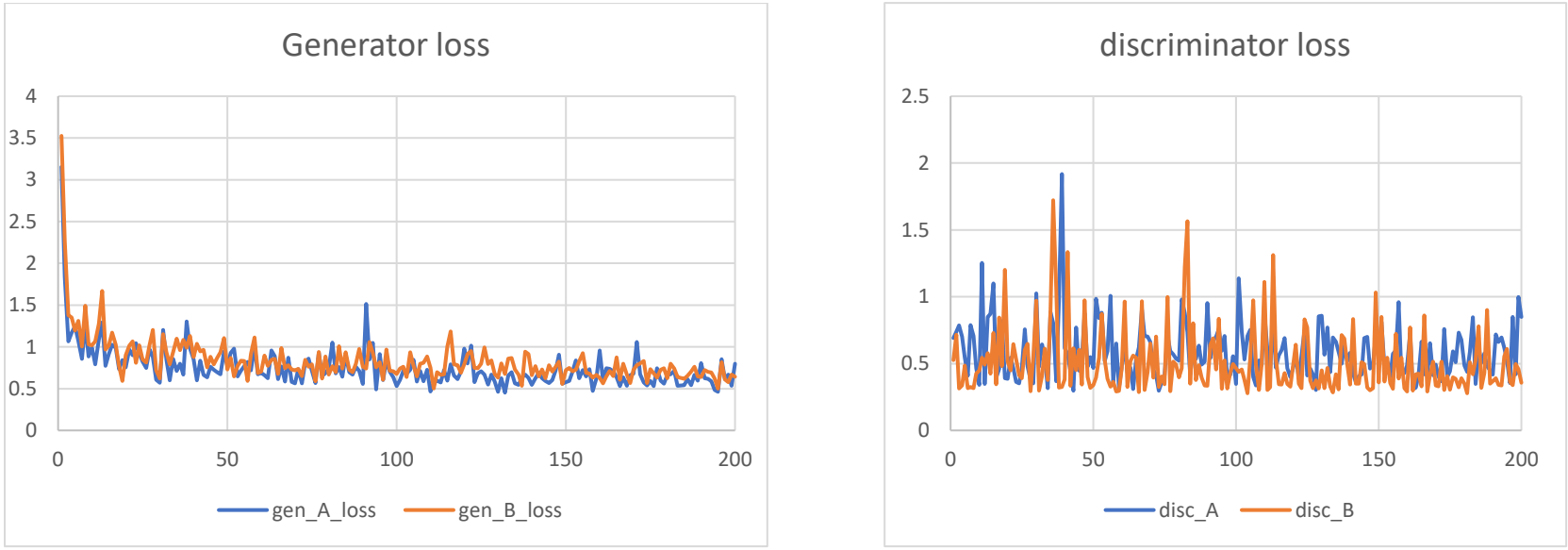
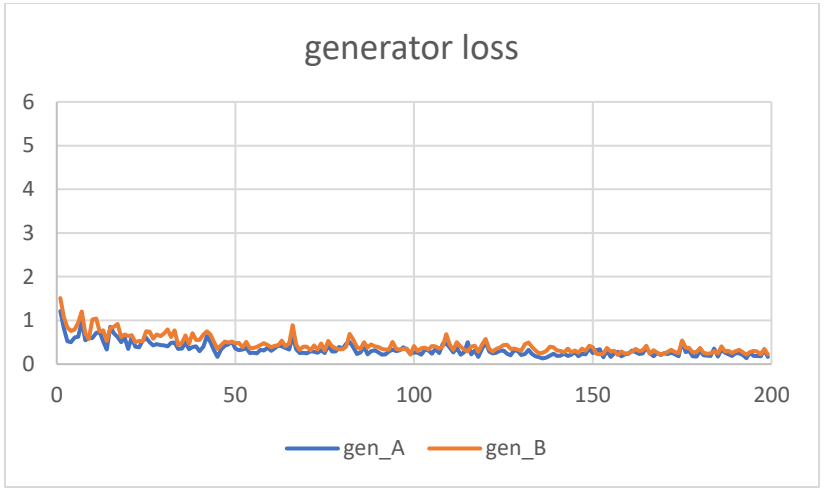


Figure A. 2: Training loss of generator and discriminator for RdUPGHIG



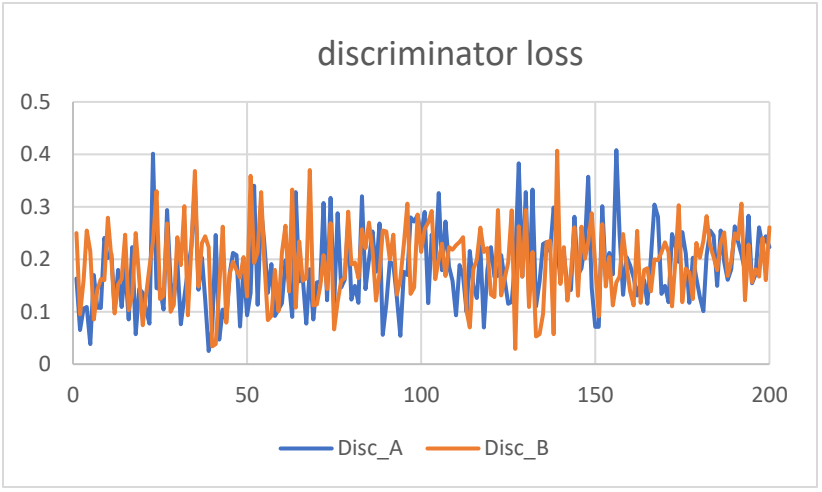
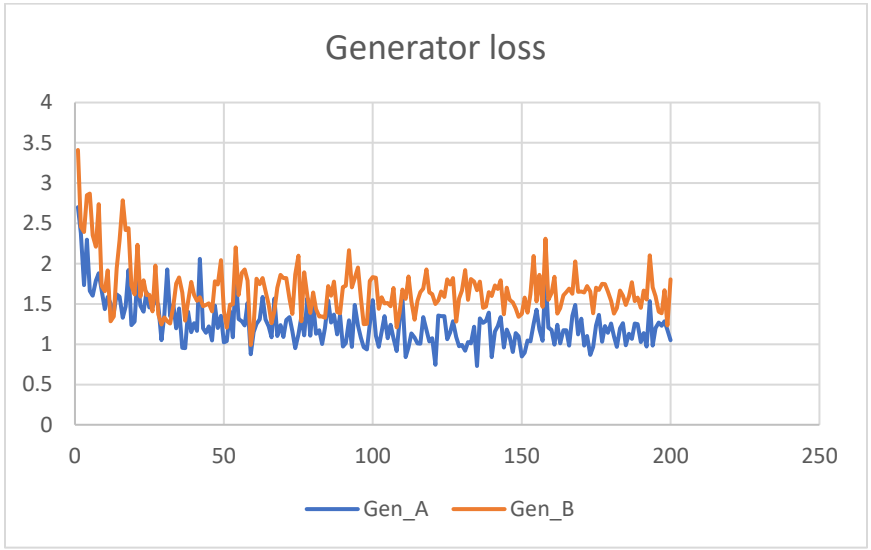


Figure A. 3: Training loss of UPG_GAN



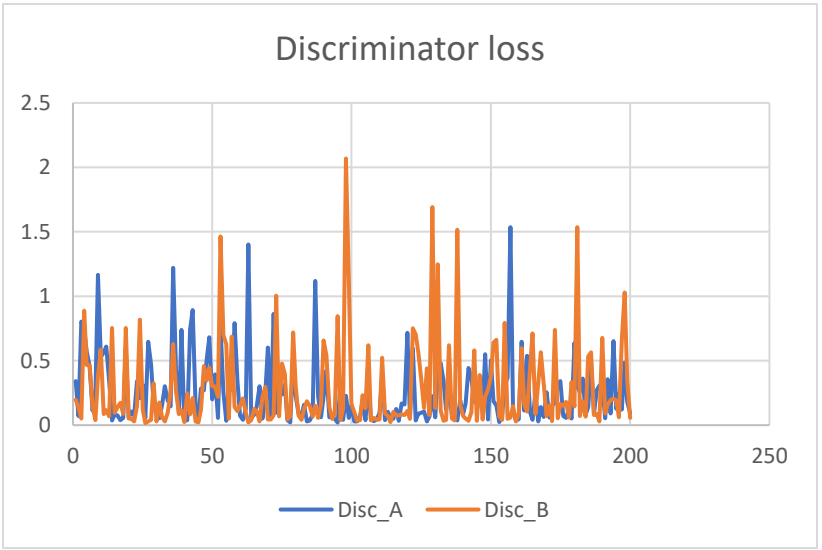


Figure A. 4: Training loss with LSGAN

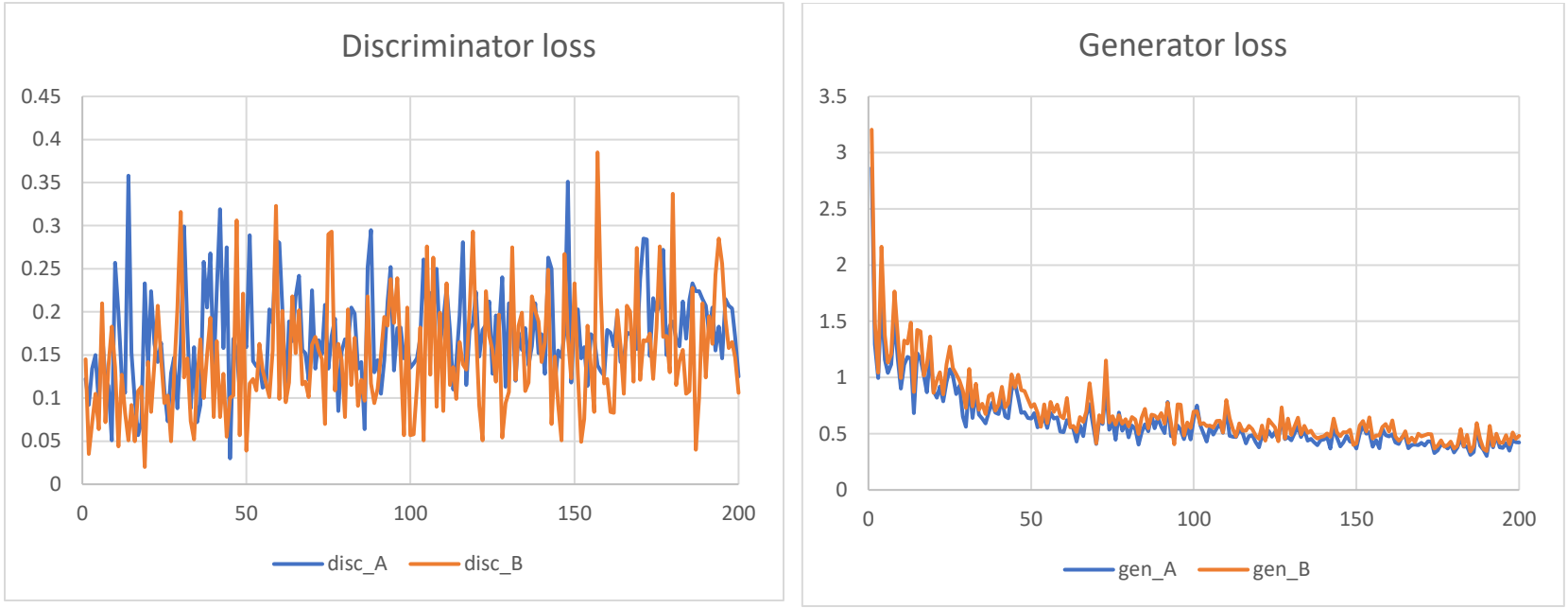


Figure A. 5: Training loss with CC+CCCL

Appendix C: Results of training at different epochs

Figure A. 6: Training results at epoch 50 and 100

Figure A. 7: Training results at epoch 150 and 200

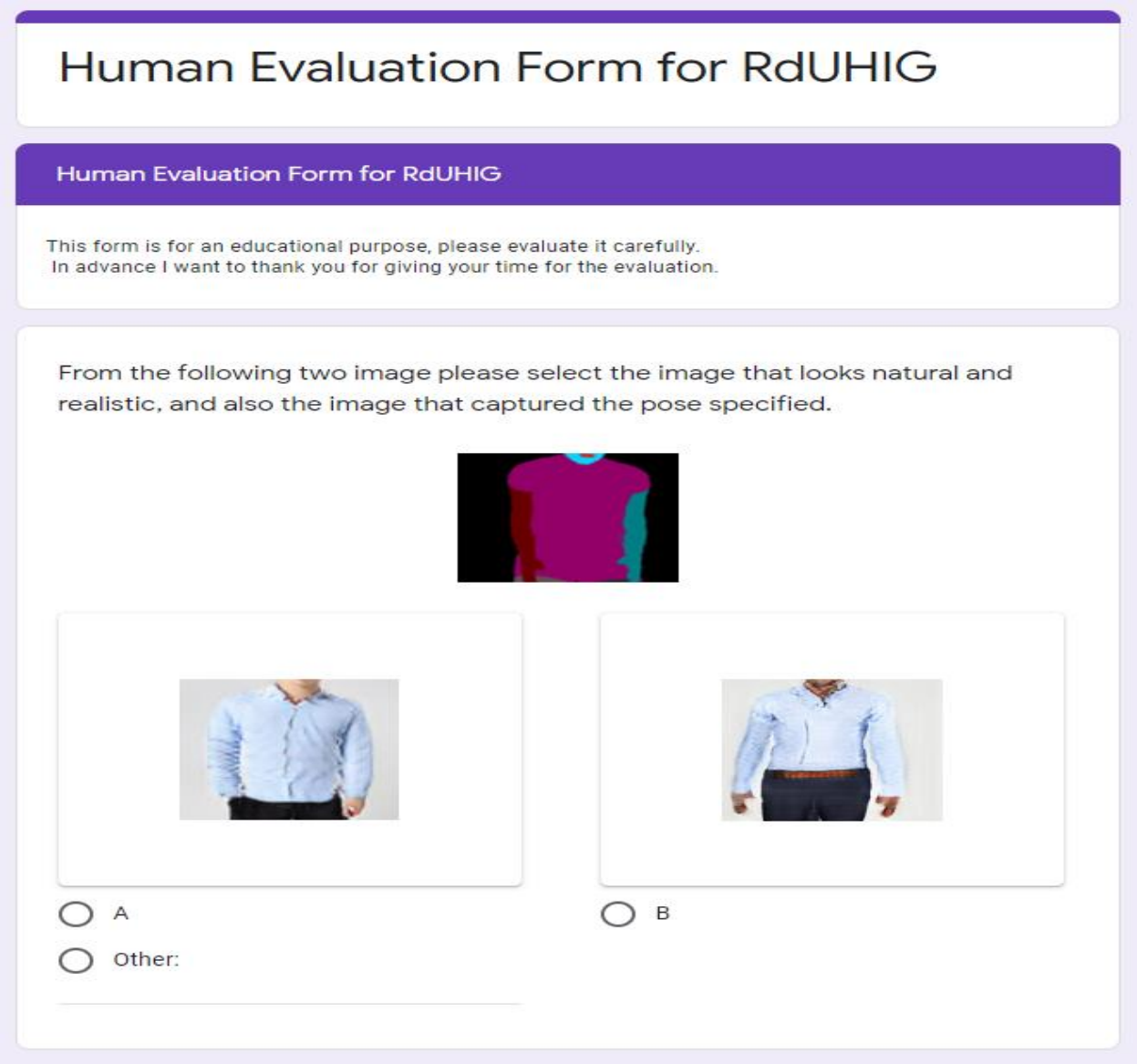
Appendix D: Sample codes of the model

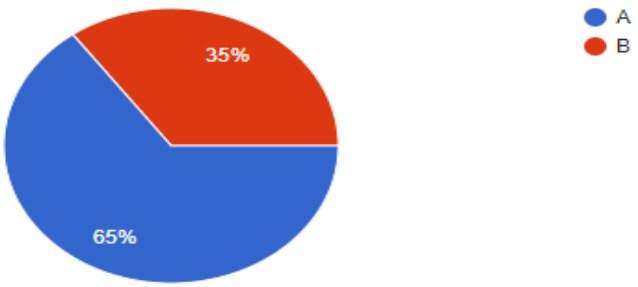
```
#network for encoder
self.netE = networks.define_E(output_channel=3, number_of_latent, nef=64,
                            which_model_netE=opt.which_model_netE,
                            norm=instance, nl=opt.nl,
                            init_type=normal, gpu_ids=self.gpu_ids,
                            vaeLike=True, pooling='max')



which_G_A  = 'resnet_6blocks_all' if opt.fineSize < 256 else
'resnet_9blocks_all'
#network for generator A and B

self.netG_A = networks.define_G(input_channel=3, output_channel=3,
                              ngf=64, which_G_A, norm=instance, opt.dropout,
init_type=normal, self.gpu_ids, nz= 17)
self.netG_B = networks.define_G(output_channel=3, input_nc=3,
                              ngf = 64, opt.which_model_netG, norm=instance,
opt.dropout, init_type= normal, self.gpu_ids)

#network for discriminators

if self.isTrain:
use_sigmoid = opt.no_lsgan
if opt.cond_D:
    self.netD_A = networks.define_D(output_channel =4, ndf =64,
                                    opt.which_model_netD,
                                    opt.n_layers_D, norm= instance,
use_sigmoid, opt.init_type, self.gpu_ids)
else:
    self.netD_A = networks.define_D(output_nc=3, ndf=64,
                                    opt.which_model_netD,
                                    opt.n_layers_D, opt.norm, use_sigmoid,
init_type= normal, self.gpu_ids)

self.netD_B = networks.define_D(input_channel, ndf,
                                opt.which_model_netD,
                                opt.n_layers_D, opt.norm, use_sigmoid,
init_type=normal, self.gpu_ids)
if not self.isTrain or opt.continue_train:
    which_epoch = opt.which_epoch
    self.load_network(self.netE  , 'E'  , which_epoch)
    self.load_network(self.netG_A, 'G_A', which_epoch)
    self.load_network(self.netG_B, 'G_B', which_epoch)
    if self.isTrain:
        self.load_network(self.netD_A, 'D_A', which_epoch)
        self.load_network(self.netD_B, 'D_B', which_epoch)

if self.isTrain:
    self.old_lr = 0.0006
    self.fake_A_pool = ImagePool(50)
```

```
    self.criterionGAN = networks.GANLoss(use_lsgan=not opt.no_lsgan,
tensor=self.Tensor)
    self.criterionCycle = torch.nn.L1Loss()
    self.criterionL1 = torch.nn.L1Loss()
    self.criterionIdt = torch.nn.L1Loss()
    self.criterionTV = networks.TVLoss()
    self.criterionClass = torch.nn.L1Loss()
    self.criterion_relativivistic_GAN = torch.nn.BCEWithLogitsLoss()

    # initialize optimizers

    self.optimizer_E = torch.optim.Adam(self.netE.parameters(), lr=opt.lr,
betas=(opt.beta1, 0.999))
    self.optimizer_G =
torch.optim.Adam(itertools.chain(self.netG_A.parameters(),
self.netG_B.parameters()),lr=opt.lr, betas=(0.5, 0.999))
    self.optimizer_D_A = torch.optim.Adam(self.netD_A.parameters(),
lr=opt.lr, betas=(0.5, 0.999))
    self.optimizer_D_B = torch.optim.Adam(self.netD_B.parameters(),
lr=opt.lr, betas=(0.5, 0.999))
    self.optimizers = []
    self.schedulers = []
    self.optimizers.append(self.optimizer_E)
    self.optimizers.append(self.optimizer_G)
    self.optimizers.append(self.optimizer_D_A)
    self.optimizers.append(self.optimizer_D_B)
    for optimizer in self.optimizers:
        self.schedulers.append(networks.get_scheduler(optimizer, opt))
```

Appendix E: Human evaluation study form and results

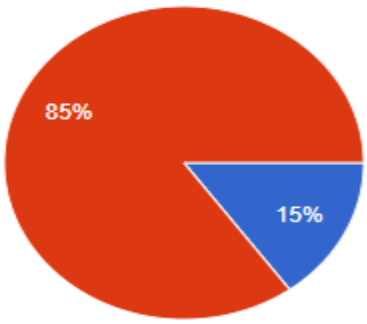
Appendix E.1 Human evaluation google form snapshots





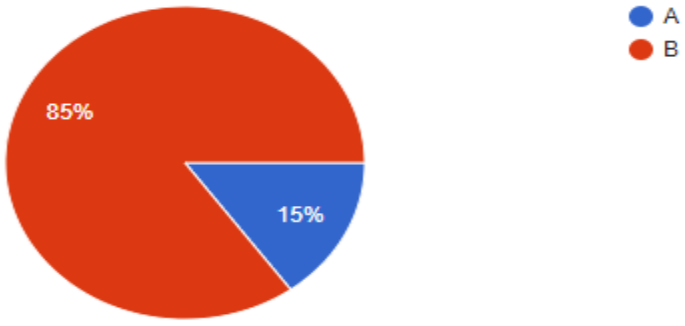
From the following two image please select the image that looks natural and realistic, and also the image that captured the pose specified.

- ○ A
- ○ B
- ○ Other:



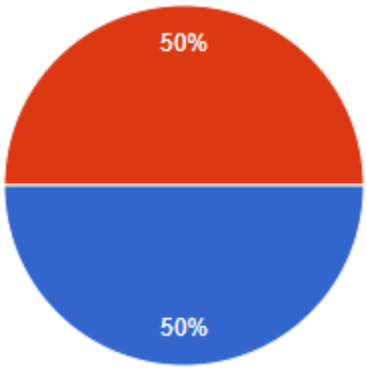
From the following two image please select the image that looks natural and realistic, and also the image that captured the pose specified.

- ◯ A
- ◯ B
- ◯ Other:



From the following two image please select the image that looks natural and realistic, and also the image that captured the pose specified.

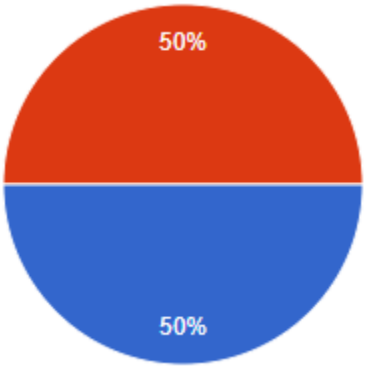
- ○ A
- ○ B
- ○ Other:



From the following two image please select the image that looks natural and realistic, and also the image that captured the pose specified.

- ◯ A
- ◯ B
- ◯ Other:

Back Submit



Appendix E.2 Human evaluation results

Table A. 1: Human evaluation results ofshirt and t-shirt

No.	Name		Shirt and t-shirt	
			Q1	Q2
1.	Tizita		A	B
2.	Adugna		A	B
3.	Mifta		A	B
4.	Dedefo		B	B
5.	Seid		A	B
6.	Mengistu		A	B
7.	Raji		A	B
8.	Kidist B		B	A
9.	Kidist A		A	B
10.	Bethlem		A	B
11.	Dagimawit		A	B
12.	Muktar		B	B
13.	Ahmed		B	B
14.	Tsegaye		B	B
15.	Ebisa		A	B
16.	Gemechis		A	B
17.	Falmiwak		A	B
18.	Anteneh		A	B
19.	Segni		B	A
20.	Mosisa		A	B
Total		A	13/20=0.65	3/20=0.15
		B	7/20=0.35	17/20=0.85

Where the results of RdUPGHIG are Q1= A and Q2=B,

the results of UPG_GAN are Q1=B and Q2 = A

Table A. 2: Human evaluation results of suit and dress

No.	Name		Suit and dress		
			Q3	Q4	Q5
1.	Tizita		B	A	A
2.	Adugna		B	A	A
3.	Mifta		A	A	A
4.	Dedefo		A	B	A
5.	Seid		B	B	B
6.	Mengistu		B	A	B
7.	Raji		B	B	B
8.	Kidist B		A	B	A
9.	Kidist A		B	A	B
10.	Bethlem		B	A	A
11.	Dagimawit		B	B	A
12.	Muktar		B	B	B
13.	Ahmed		B	B	A
14.	Tsegaye		B	B	A
15.	Ebisa		B	A	B
16.	Gemechis		B	A	B
17.	Falmiwak		B	A	A
18.	Anteneh		B	B	B
19.	Segni		B	B	B
20.	Mosisa		B	A	B
Total		A	3/20=0.15	10/20=0.5	10/20=0.5
		B	17/20=0.85	10/20=0.5	10/20=0.5

Where the results of RdUPGHIG are Q3= B and Q4=A, Q5=A

the results of UPG_GAN are Q3= A and Q4=B, Q5=B